



Serhiy Moskovchuk

Nº 42010

Video Metadata Extraction in a VideoMail System

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática

Orientador : Doutor Nuno Manuel Robalo Correia, Prof. Catedrático,
Faculdade de Ciências e Tecnologia da UNL - Departa-
mento de Informática

Júri:

Presidente: Doutor Pedro Manuel Corrêa Calvente Barahona,
Prof. Catedrático, Faculdade de Ciências e Tecnologia da UNL
Departamento de Informática

Arguente: Doutora Maria Teresa Caeiro Chambel, Prof^a Auxiliar, Faculdade de
Ciências, Universidade de Lisboa - Departamento de Informática

Vogal: Doutor Nuno Manuel Robalo Correia, Prof. Catedrático, Faculdade de
Ciências e Tecnologia da UNL - Departamento de Informática



FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

May, 2015

Video Metadata Extraction in a VideoMail System

Copyright © Serhiy Moskovchuk, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa

A Faculdade de Ciências e Tecnologia e a Universidade Nova de Lisboa têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objectivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

To my family. Obviously.

Acknowledgements

First of all I wish to express my sincere thanks to my advisor Dt. Nuno Manuel Robalo Correia, for providing me with all the necessary facilities for the research, scientific guidance, participation in the discussions, continuous support and of course his enormous patience.

I am also grateful to the Faculty of Science and Technology, New University of Lisbon and the WeWoW company for providing working conditions, required materials and facilities. Specially to WeWoW for the creation of this project and the team who supported me all the time.

In addition, a thank to my friends and colleagues André Grossinho, Pedro Santos, Rui Madeira, André Sabino, Flávio Martins, André Mourão, Bruno Cardoso, Ana Figueiras, Filipa Peleja and Inês Rodolfo for helping me during the work and for everything else.

Finally, thanks to all other people around who directly or indirectly, purposely or not influenced this work.

Abstract

Currently the world swiftly adapts to visual communication. Online services like YouTube and Vine show that video is no longer the domain of broadcast television only. Video is used for different purposes like entertainment, information, education or communication.

The rapid growth of today's video archives with sparsely available editorial data creates a big problem of its retrieval. The humans see a video like a complex interplay of cognitive concepts. As a result there is a need to build a bridge between numeric values and semantic concepts. This establishes a connection that will facilitate videos' retrieval by humans.

The critical aspect of this bridge is video annotation. The process could be done manually or automatically. Manual annotation is very tedious, subjective and expensive. Therefore automatic annotation is being actively studied.

In this thesis we focus on the multimedia content automatic annotation. Namely the use of analysis techniques for information retrieval allowing to automatically extract metadata from video in a videomail system. Furthermore the identification of text, people, actions, spaces, objects, including animals and plants.

Hence it will be possible to align multimedia content with the text presented in the email message and the creation of applications for semantic video database indexing and retrieving.

Keywords: Media annotation, Media labeling, Video analysis, Object identification, People identification, Text segmentation, Face detection and recognition

Resumo

Atualmente o mundo rapidamente adapta-se à comunicação visual. Serviços online como YouTube e Vine mostram que o vídeo já não é somente de domínio da TV. O vídeo começou a ser usado para objetivos diferentes como o entretenimento, informação, educação ou comunicação.

O crescimento rápido dos arquivos de vídeo atuais com dados parciais oferecidos pelo editor cria o grande problema da sua obtenção. Os humanos olham o vídeo como um conjunto complexo de conceitos cognitivos. O resultado disto é a necessidade de construir a ponte entre valores numéricos e conceitos semânticos. Isso estabelecerá uma ligação permitindo obtenção fácil dos mesmos.

O aspeto crítico desta ponte é anotação do vídeo. O processo de anotação pode ser realizado manualmente ou automaticamente. A anotação manual é muita demorada, subjetiva e cara. Assim a anotação automática está a ser ativamente estudada.

Este trabalho foca-se em anotação automática dos conteúdos multimédia, especialmente no uso das técnicas de análise para obtenção da informação possibilitando extração automática dos metadados do vídeo num sistema videomail. Foi assim realizada identificação do texto, pessoas, espaços e objetos.

Assim será possível alinhar o conteúdo multimedia com o texto presente na mensagem email e a criação das aplicações para indexação e acesso semântico à base de dados.

Palavras-chave: Anotação multimédia, Análise de vídeo, Identificação dos objectos, Identificação das pessoas, Segmentação de texto, Detecção e Reconhecimento facial

Contents

1	Introduction	1
1.1	Motivation	2
1.2	Problem Description	2
1.3	Realized Solution	4
1.4	Main Contributions	4
1.5	Document Organization	5
2	Related Work	7
2.1	Video Annotation Systems	8
2.1.1	The basics of the automatic annotation process	8
2.1.2	AV Portal of the German National Library of Science and Technology	13
2.1.3	INHA Video Annotation Tool	15
2.1.4	ViewProcFlow Annotation Tool	15
2.2	Scene Text Detection and Recognition	19
2.2.1	Text Localization Techniques	20
2.2.2	Optical Character Recognition	33
2.3	People Identification and Recognition	39
2.4	Concepts and Object detection	46
2.5	Action Identification and Recognition	49
2.5.1	Single-layered Approaches	49
2.5.2	Hierarchical Approaches	51
2.5.3	Existing Methods	52
3	Solution	55
3.1	Design	56
3.1.1	Use cases	56
3.1.2	VeedMee architecture	58
3.1.3	VeedMind architecture	60
3.2	Implementation	64

3.3	Results	71
3.3.1	Discussion	71
3.3.2	Performance	72
4	Conclusions and Future Work	75
4.1	Conclusions	75
4.2	Future Work	76

List of Figures

1.1	Example of VeedMee’s user interface	3
1.2	Diagram of VeedMee’s communication process workflow	3
2.1	Image representation in computer vision	7
2.2	Color features	10
2.3	Visual features 1	10
2.4	Visual features 2	11
2.5	INHA-VAT Architecture	16
2.6	VideoProcFlow Architecture	18
2.7	C. Shi’s method workflow	22
2.8	Flow chart of the [113] method	24
2.9	MSER bounding boxes from three channels	25
2.10	Q. Ye’s method workflow	26
2.11	Neumann’s component tree	27
2.12	Flow chart of the [22] SWT method	28
2.13	Gradient Vector Flow examples	29
2.14	Tesseract OCR baselines	37
2.15	Tesseract spacing example	38
2.16	Haar features	40
2.17	Face detection process workflow	40
2.18	Integral images examples	46
2.19	SURF keypoints examples	47
2.20	SURF descriptor	48
2.21	Human action recognition approach-based taxonomy	50
2.22	The classification accuracies of human action detection systems tested on the KTH dataset.	53
3.1	VeedMind use case diagram	57
3.2	VeedMind integration architecture	59

3.3	VeedMind component diagramm	61
3.4	Face detection and normalization	62
3.5	Text detection and normalization	63
3.6	VeedMind workflow diagram	65
3.7	Segmentation XML file	68
3.8	Final XML file	69

List of Tables

2.1	Performance comparison of the C.Shi et al. method (%)	21
2.2	Performance comparison of Q.Ye et al. 2005 method	24
2.3	Performance comparison of the Q.Ye et al. 2014 method (%)	26
2.4	Performance comparison of the SWT algorithm	29
2.5	Performance comparison of the GVF algorithm	30
2.6	Performance comparison of P. Shivakumara et al. method in (%)	31
2.7	Test recognition accuracy on ICDAR 2003 character sets in (%)	35
2.8	Word recognition rates of C. Shi et al. method	36
3.1	Segmentation results of testing videos	71
3.2	Concept detection results	72
3.3	Task testing times	73

Listings

3.1	Produced Example XML	70
-----	--------------------------------	----



Introduction

Communication is the process of conveying information. It started with spoken word stories, evolved through cave paintings, carrier pigeons, morse code, telephone, email, instant messaging, social media and networking. The main reason of this evolution is a technological progress. Every innovation brings some changes or even new abilities to communication.

The telephone extended geographical limits of verbal communication, computers brought non-verbal communication into the digital era improving its temporal disadvantages. Smartphones made the full spectrum of verbal and non-verbal communication and even visual. As a result technology has brought more communication options.

All these different communications need some channels to work. The Internet is one of these main channels and now it overpassed three billions of users [93]. It offers many services and one of them is an electronic mail system. Its appearance gave people a new way to communicate which was quickly accepted and became widely used. Mail systems grew to tremendous number of users and now it is being projected to 2.5 billion users [28]. Such number of users generates a huge amount of data and that is an increasing problem as people are more connected to email through the pervasiveness of smart phones and tablets, the ever-increasing availability of Internet connectivity, and the wider spread of social networking [73]. Email is present in all facets of daily life: personal (communication with friends and family, manage bills and juggle between groups and activities) and work (communication between coworkers) and there is a need of understanding its behavior and deal with it [26].

VeedMee is a new communication tool that goes beyond email technology. It adds up all the advantages of audio and video. It is possible to record your videos on your computer or smartphone and send them. This is a new way of communication by email,

the closest to be physically present, either in a meeting with your co-workers or loved ones [99].

VeedMee is an email system that uses video to communicate and this leads to the aim of our project: developing techniques for multimedia information search to offer a possibility of multimedia content alignment with the text presented in the email message.

1.1 Motivation

Today we observe a huge increase of mobile devices, capable to receive and transmit large amounts of data. The number of internet access points and the capability of the network is also increasing. The amount of internet traffic coming from non-computer devices is quite significant and is growing [73].

The system we are dealing with, videomail, has support on computers and smart-phones, so we must expect a huge amount of data to deal with, namely video. This results in a problem of overloading the user with it. The study made by Topkara et al [96] showed us that email messages including thumbnails drove significantly higher the click through rates and surveys show that users found thumbnail views more appealing and novel. Apart from this we can reach better results even, offering more information about video: who appears, where it is filmed, what objects are present. This information will help the user to better management of their videomails, allowing to select the most suitable materials and finding them faster.

1.2 Problem Description

This work is integrated within the project VeedMind, in partnership with the company WeWoW, funded by QREN (*Quadro de Referência Estratégico Nacional*), with the aim of developing multimedia information search capabilities in the videomail system VeedMee.

Like other standard email systems, VeedMee probably will suffer the same problems, namely email overload, that means receiving a large volume of incoming email and having emails of different status types (to do, to read, etc) [26]. In our case a large volume of incoming email is even more difficult, since we deal with text and multimedia data. The huge amount of incoming correspondence will complicate the user's life, so it is crucial to offer the possibility of searching within the video content and align it with the email's text content.

The VeedMee system is designed to be simple to learn and to use. Figure 1.1 shows the user interface that is very similar to other widely used email clients. The workflow of creating and sending *veedmees* is also quite simple. It is described in Figure 1.2 that shows a diagram of the process.

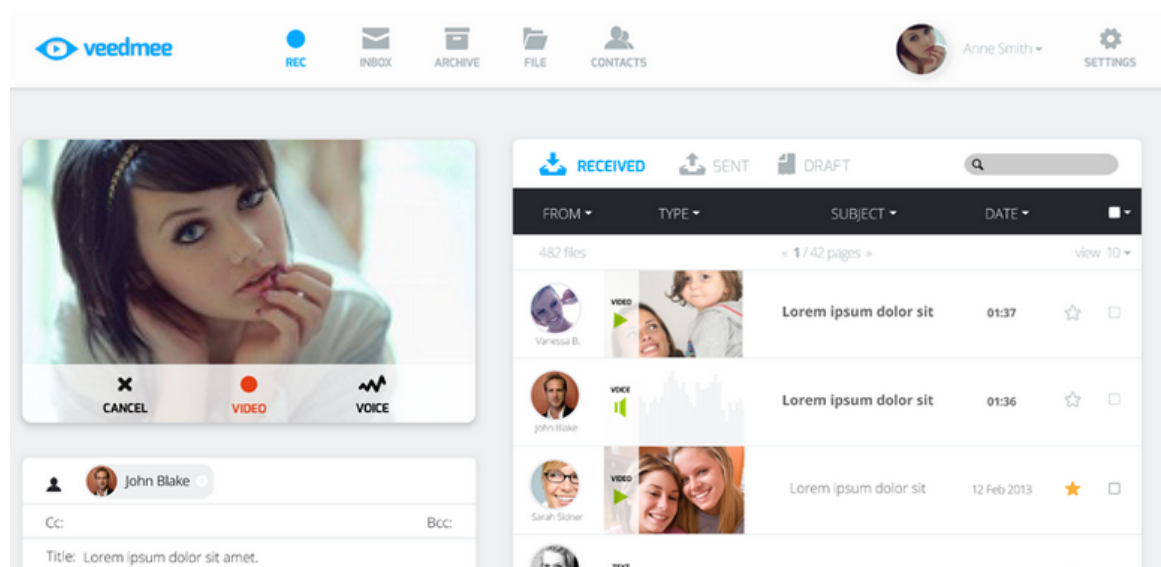


Figure 1.1: Example of VeedMee's user interface

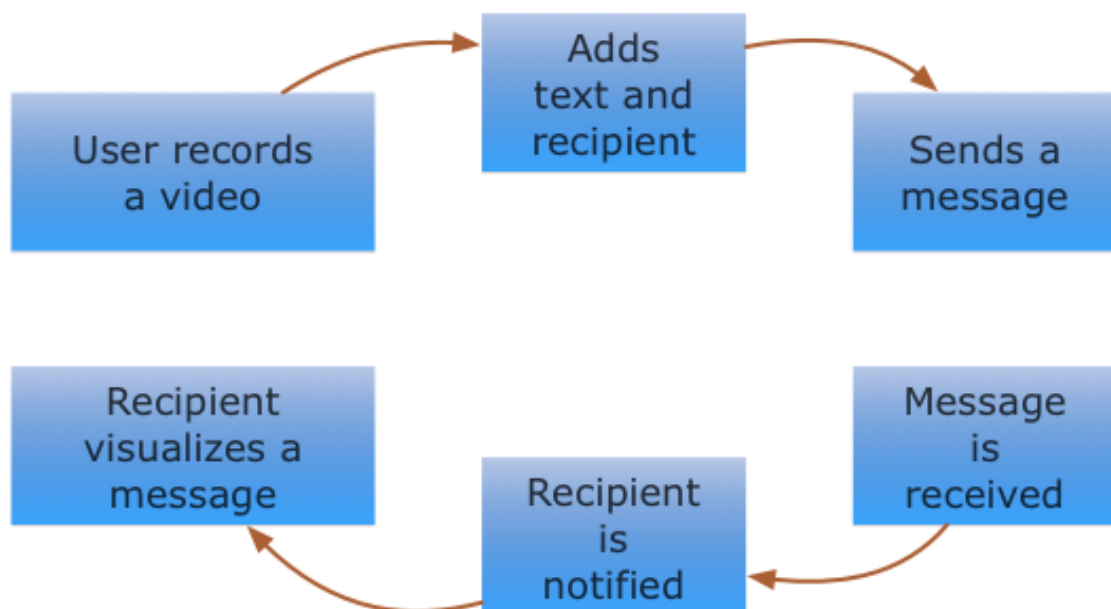


Figure 1.2: Diagram of VeedMee's communication process workflow

1.3 Realized Solution

The realized solution consists in the use of techniques that automatically extract metadata from video, namely: the identification of text, people, spaces, objects.

Videos are stored in MP4 format, that is widely used and is supported by many devices. The system gets a video and analyzes it in the following order:

1. Segmentation - a process of detecting cuts. The video is transformed into a list of key-frames, each key-frame characterizes a part of video from what it is cut.
2. Face detection - the system checks a presence of human faces. If those are found it stores information about them.
3. Face recognition - having a list of frames where human faces are present, the system tries to recognize those faces.
4. Text detection - each key-frame is checked for a presence of text, if the text is found, it is registered in the system for future recognition.
5. Text recognition - the system recognizes previously detected text. The Tesseract OCR is employed for that.
6. Object detection - a feature descriptor is extracted from each key-frame. Then they are stored for future searching process.
7. Concept detection - the system has a set of previously trained concepts like: beach, car, airplane. Each key-frame is checked for a presence of those concepts.

Extracted metadata will be integrated with the VeedMee database, so it could be used in the future to offer users new functionalities like searching the video content, describe video before it is visualized and marketing.

1.4 Main Contributions

The realized solution offers an efficient way of exacting metadata using existing state of the art algorithms and libraries, namely OpenCV and openFrameworks. The results will be integrated in the VeedMee videomail system. The realized solution showed its performance. As a result, it offers a good set of tools for metadata extraction. It is separated into modules, which can be used separately and for different purposes.

1.5 Document Organization

In addition to this introductory chapter, the remainder of the document is organized as follows:

- **Chapter 2:** Related Work overview addressing the topics of:
 - Video Annotation Systems (Section 2.1)
Review of existing video annotation systems.
 - Scene Text Detection and Recognition (Section 2.2)
Existing techniques of text detection and its recognition with OCR.
 - People Identification and Recognition (Section 2.3)
Explains how people are detected and recognized by computer.
 - Concepts and Object detection (Section 2.4)
Discusses concepts and object detection on images (video frames).
 - Action Identification and Recognition (Section 2.5)
Discusses human actions/activities recognition techniques.
- **Chapter 3:** Solution, presentation of realized solution:
 - Design (Section 3.1)
Explains the design of the realized solution.
 - Implementation (Section 3.2)
Describes a solution implementation.
 - Results (Section 3.3)
Discusses tests and results on the realized system.
- **Chapter 4:** Conclusions and Future Work:
 - Conclusions (Section 4.1)
Presents final conclusions of this work.
 - Future Work (Section 4.2)
Discusses future improvements of realized work and solution.

Related Work

This dissertation focuses on multimedia material manipulation. As a result it is related to the area of Computer Vision. Authors of [39] define *Computer Vision* by the transformation of data from 2D/3D stills or videos into either a decision or a new representation, that is done for achieving some particular goal. In a machine vision system, a computer receives an array of numbers from the camera or from disk and this is the type of data that we deal with, Figure 2.1. As a result our task becomes to turn this array of numbers into the perception we need.

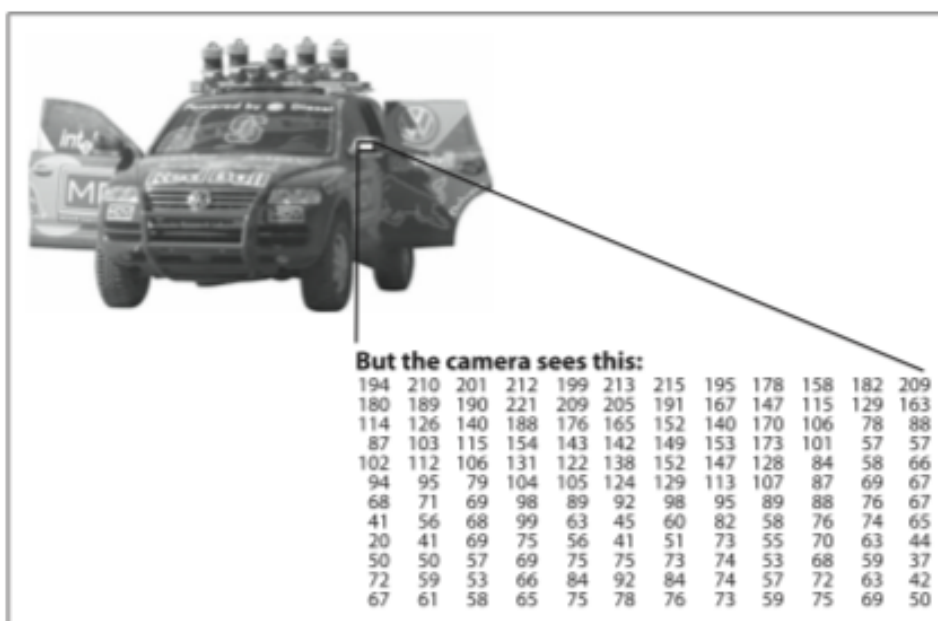


Figure 2.1: Image representation in computer vision

Hence the main focus of this chapter is to explain some concepts, present previously developed projects and studies relevant to this dissertation. It is divided into five sections:

Video Annotation Systems

Review of existent video annotation systems, as well as, used techniques for multimedia metadata extraction.

Scene Text Detection and Recognition

Existing techniques and methods to detect text localization within a scene.

People Identification and Recognition

Review of face detection and recognition approaches, its classification, benefits and drawbacks.

Concepts and Object Detection

Fast review of the existing techniques for object and concept automatic detection.

Action Identification and Recognition

Classification and discussion of existing approaches and methods for action detection in videos.

2.1 Video Annotation Systems

The rapid growth of today's video archives with sparsely available editorial data demands some work to facilitate its retrieval. Facilities to access to the complete video documents are already implemented and work well, a good example of it is YouTube. However, a solution for retrieval of specific segments are still in a nascent stage.

To provide specific content retrieval, video should be semantically perceived by the computer. This process requires labeling (annotation), where specific frames of video are semantically marked with the presence of known semantic concepts. Two types of semantic annotation are used: manual, human made and automatic (machine-driven) [92].

Instead of expensive and ambiguous manual annotation, we should process to the automatic annotation tools. As a result an automatic metadata extraction becomes a crucial ingredient for semantic-level video browsing, search and navigation. Subsection below explains the basics of the automatic annotation process.

2.1.1 The basics of the automatic annotation process

In general the automatic annotation process starts with a segmentation phase. As we know, video is a sequence of images (frames). These sequences of frames could have relations between them. The segmentation phase searches for these relations and divides video in independent segments. Highly robust automatic methods exist [117]. All of

them compare successive frames, filtering them with some static or dynamic threshold. Various techniques such as pixel-based, histogram, edge, motion and even mean and deviation intensities have been proposed. As a result the segmentation process yields a list of key-frames. In this context key-frames are the frames that represent a related sequence segment.

Then a concept-detection phase is employed. Essentially it is a machine learning problem. Given an n -dimensional feature vector x_i of a key-frame i , the aim is to obtain a measure, which indicates whether semantic concept w_j is present in shot i [92]. Basically there are three stages: feature extraction, concept learning and classification. This sequence of stages builds an actual bridge that solves a semantic gap problem, the lack of connection between digital data and semantic concepts.

The aim of feature extraction is to take an original key-frame and extract a descriptive representation of the pattern of interest, computing an n -dimensional feature vector. The last one could be described using text, audio and visual features or their combination. In this thesis we focus only on visual features. These features have different illumination variations, different viewpoints. As a result a sensory gap arises, the gap between the object in the world and the information in a computational description derived from a scene [91]. Thus there is a need to find some type of invariance that will establish solid features, insensitive to changes caused by visual properties. C. Snoek and M. Worring consider visual features along two dimensions: their type (color, texture and shape) and their spatial scale on which they are computed (global, region, key-point and temporal extensions) [92]. Below we will discuss some principles related with each one.

Color: Each frame originally is represented by an array of pixels, where each element has a color. Color could be represented in various modes, the most known are: RGB, HSV, $L^*a^*b^*$ and Black and White. The first one, is a additive color model which combines Red, Green and Blue to represent the other colors. Generally in this model, a pixel is represented by three 8-bit numbers in a range of [0-255]. Figure 2.2(a) depicts it. The HSV indicates how much certain wavelength is present. It has three components: *Hue*, the dominant wavelength in the color spectrum, *Saturation*, an amount of white and *Value*, an intensity of the color. Figure 2.2(b) shows it. The $L^*a^*b^*$, Lab color space, is based on the Opponent-Color Theory¹ and includes all perceivable colors. L^* is similar to the V in HSV, in other words: Lightness, a^* and b^* represent redness or greenness and yellowness or blueness respectively [71]. Figure 2.2(c) illustrates it.

Texture: Instead of color features, texture relies on a local groups of pixels and could be defined as an organized pattern of quite regular sub-elements. It is a very useful feature in detection of repetitive pattern concepts like: a brick wall, forest or zebra. There are statistical approaches [33] or more known methods like an application of Gabor filters [34] and wavelet features [61]. The combination of texture and color properties can deliver good results in general concept detection, for example: indoor vs outdoor, beach,

¹This theory assumes that the receptors in the human eye perceive color as pairs of opposites (Light vs dark, Red vs green and Yellow vs blue)

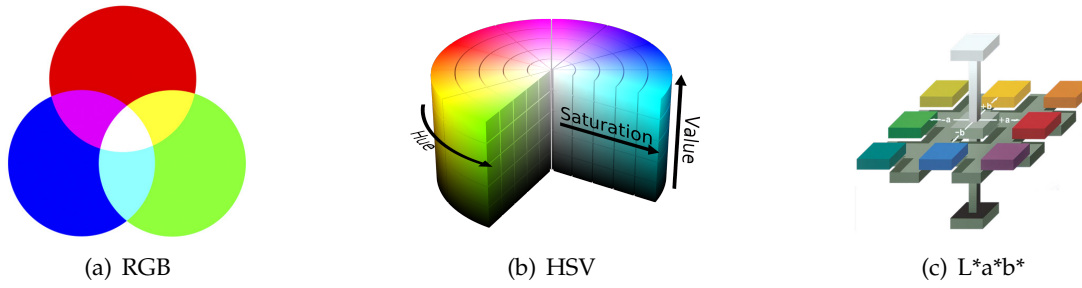


Figure 2.2: Color features

forest. Figure 2.3(b) shows an example of texture.

Shape: It is a set of connected pixels that share a specific property, V . Two pixels, p and q , are connected if there is a path from p to q of pixels with property V . A path is an ordered sequence of pixels such that any two adjacent pixels in the sequence are neighbors [78]. Figure 2.3(a) depicts an example. Connected component could be characterized by moments, including an area, centroid or orientation of the region. These moments are related to computational geometry, the subarea of algorithms design that deals with the design and analysis of algorithms for geometric problems, threads this feature extraction and matching problems, a good review is made in [100]. These features are good, but they have some drawbacks due to their direct visual data dependence when changing a point of view or a scene extremal changes.



Figure 2.3: Visual features 1

Global: These type of features works directly on a full frame and generally represented as histograms. The basic approach counts a number of pixels with a certain color. It is a very simple way to characterize a frame but it irreversible. Histograms loose a localization information of pixels. Histograms are a good way to detect scene changes. In 2005, J. Geusebroek and A. Smeulders showed that the complete range of image statistics in natural textures can be well-modeled with an integrated Weibull probability distribution [23]. Hence a complete histogram may be reduced to distribution parameters. Another approaches like a color coherence vectors [74], color correlograms [30] and Markov stationary features [50] are alternative representation which conserve spatial neighborhood information and improve retrieval performance. An example histogram is shown

in Figure 2.4(a).

Region: We can split a frame into regular partitions, called regions. This partitioning process could use different combination of rectangles or even just one. J. Van et al. divide an image in overlapping regions which are uniformly sampled across the image, with a step size of half a region [98]. Example of regions could be seen in Figure 2.4(b).

Key-point: This is a local type feature. A local feature is an image pattern which differs from its immediate neighborhood [97]. Generally we work with them in a form of key-points: the information-rich pixels in the image. But they can be represented as edges, corners or junctions. A good study of local invariant feature detectors is made in [97] and [82]. Also there is a large study of a combination with codebooks [107], [13], [51]. Figure 2.4(c) depicts an example.



Figure 2.4: Visual features 2

Temporal: Since we work with video data, that is an ordered sequence of frames, we should consider it's temporal properties. C. Snoek references some examples: analyzing motion pattern to find the camera motion, track regions or points of interest through the sequence and describe their tracks, derive general measures of the activity in the scene [92]. Another study for event detection and recognition for semantic annotation of video where temporal features play a big role is made by L. Ballan et al. [7].

There is a an ISO/IEC standard developed by Moving Pictures Expert Group for metadata representation: MPEG-7 [14]. MPEG Group aims to define a standard for set of descriptors that could be used to describe various types of multimedia information. MPEG-7 already contains a set of audio-visual descriptors, as well as, ways to define other descriptors, structures and relations between. J. Laaksonen et al. present a system for an image retrieval that uses the MPEG-7 features [44].

The next step after the feature extraction is classification. Classification will yield an existence or an absence of a concept. It is divided into two big phases: Training (Learning) and Classification. As the name describes, the first phase will train a classifier for a semantic concept w_j and a second indicates whether w_j is present in shot i . Both, training and classification, demand a feature extraction, computing of an n -dimensional feature vector. Thus, each feature serves as a unit of measure of an image. To improve computational demands, scientists from statistics and machine learning area developed methods that reduce the computational weight of the features. M.Jordan and J. Kleinberg presented a review of those [38].

Classifiers training methods have two main approaches: *Supervised* and *Unsupervised* learning. The first one is highly used in concept detection stages of semantic annotation systems that we are discussing here. Hence we will discuss it in more details. The main difference between these two approaches is a training data labeling. Supervised learning needs labeled data, it is marked as positive or negative examples (true or false). Unsupervised approaches receive a big amount of unlabeled data and then try to find clusters (groups) with similar features. This type of learning is very computationally expensive and demand much time to yield state of the art results [15]. Quoc V. Le et al. investigate the feasibility of building high-level features from only unlabeled data. Experimental results confirm that it is possible and show that the learned detector is invariant to translation and even to out-of-plane rotation and scaling [46].

S. Kotsiantis et al. define that a goal of supervised learning is to build a concise model of the distribution of class labels in terms of predictor features. The resulting classifier is then used to assign class labels to the testing instances where the values of the predictor feature are known, but the value of the class label is unknown [42]. Also S. Kotsiantis makes an overview of Supervised Machine Learning Classification Techniques in the same paper. In other words, the goal of supervised learning is to define a model, used by the classifier, which will indicate the learned label of unknown input concept basing on extracted features. In order to train a good classifier, training examples should be balanced as well as a number of used features [33]. The most known supervised learning machine in this area is a Support vector machine.

A *Support Vector Machine*, *SVM*, is a learning method used for binary classification. It aims to find a hyperplane which separates the d -dimensional data perfectly into its two classes. Since the training data is often not linearly separable, *SVM* introduces a notion of "kernel induced feature space" which transforms the data from d -dimensional input vector into a (usually higher) d' -dimensional vector. Thus it is allocated in a higher dimensional space, where it turns separable [11]. To improve classification of the *SVM* classifier, we should provide a good parametrization. There are two parameters: C and K . C indicates an optimization of a hyperplane of a machine when $C = \infty$, the algorithm tries to completely separate data, for a finite values, the problem reduces into finding a "soft-margin" classifier. The K , *Kernel function* is a mapping function that indicates how to separate data into a higher dimensional space. Since each case is different, different parameters are needed. Thus, a standard way of selection is an iterative search on a large number parameters values selecting the best whose yielded better results.

Concept classification in general yields binary results. Guo-Jun Qi et al. reports that researches in this field evolved through two paradigms [77]. The first one threads each concept individually, detecting it by binary classifiers. However, it ignores relations between them. As a result, the second one evolves, adding an extra step on the top of the first individual classifiers to fuse the multiple detections of the concepts. Another

paradigm is a merge of the first two ones. It simultaneously models both the individual concepts and their correlations in a unifying formulation. It is also proposed in paper [77]. Another way to improve video annotation is to merge metadata from different sources, for example speech transcript and the visual content, called multimodal fusion. S. Ayache et al. present in a study and comparison between three fusion schemes. The first merges collected features before classification, second, adds a normalization process which balances features weights, the last one, performs a fusion at a concept level, after classification [6]. The study concluded that any fusion presented above perform in average better. Selection of the scheme depends of a nature of the concepts.

After a classification phase, video annotation systems enable a content retrieval which is the main propose. Thus, system should return the most relevant material queried by user. User can demand a different type of data (entire video, frame, temporal instant or tags) and a different type of querying it. Hence queries could be separated into types: query-by-keyword (searches an associated text around video, like *YouTube*), query-by-example (searches for similarities, like *Google* search by image) and query-by-concept (searches for identified concepts from classification phase). Querying mechanism could be very complex. It can combine results from different sources as well as interpret user query and transform it into another listing of words (text mining). At the end results are presented to the user. We do not discuss querying methods and user interface because this dissertation focuses only on extraction of semantic data from the video.

2.1.2 AV Portal of the German National Library of Science and Technology

This video annotation system was designed for an automatic annotation of scientific video material based on visual concepts. The need was created by the rapid growth of video archives and difficulties associated to its retrieval. A lack of tools to search for a specific visual concepts semantically was identified in studies during design stage of the AV Portal provided by German National Library of Science and Technology.

In 2013, the German National Library of Science and Technology holds approximately 2,000 hours of video data with an approximated growth of 1,000 hours every year [29]. This multimedia material is not indexed as standard textual data. Thus a major focus of this project was the exploitation of techniques for automatic indexing of the visual content depicted in video recordings. Thus it will be possible provide to user a semantic search and specific content retrieval.

As we mentioned before, the video archive is huge and includes multimedia material from a vast number of scientific areas. Thus, designers created categories and made a survey to find a specific visual categories interesting to the user in specific category. Together with subject experts and survey results subject-specific and cross-subject concepts were established. Also label names were aligned with labels provided by the German Authority File *Gemeinsame Normdatei* (GND) used for cataloging in library context.

In relation to the training data, it was manually selected as a single video key-frames.

Each key-frames was selected and described by the subject expert in order to guarantee best quality of training data and concept mutual exclusion. The process started with a selection of 80 different videos per subject with varying time. As a result an average amount of 100 key frames per visual concept have been annotated. Obtained training data then feeds the Support Vector Machine classifier. Key-frames which depict a concept are labeled as positive, alternatively, negative are obtained by assembling all key frames of the respective subject that do not depict the concept.

The content-based classification stage begins with the segmentation, which is based on successive frames comparison on luminance histograms and edge energy features. The middle frame of each segment is selected as a key-frame. Feature extraction is characterized by the SIFT (Scale-Invariant-Feature-Transform, [57]) features at a fixed grid 6×6 pixels on each channel of a key frame in RGB color space. Thus, extraction yields a 384-dimensional feature vector at each grid point. Then a visual vocabulary, a Bag-of-Words representation of a key-frame, is computed via k-means clustering. $k = 4,0000$ cluster were previously computed from a particular subject. By assigning each feature vector of a key-frame to its most similar cluster center, a normalized histogram (Bag-of-Words) of the key-frame is computed. This combination becomes invariant to visual distortions like rotation, occlusion, lighting and intra-class variation but rich enough to carry enough information to be discriminative at the category level [18]. Computed Bag-of-Words is used as a feature vector of a key-frame used as input into classifier. In this system, a standard Kernel-based Support Vector Machine is employed. Selected parameters of machine were a Gaussian kernel based on the χ^2 distance with a $\gamma =$ average distance between all training key-frames BoW-histograms. The classification task follows a one-against-all approach, with one SVM per concept trained to solve a binary classification problem.

Accordingly to the context of the system, evaluation was done with the subject-specific testing data. Evaluation showed that not all cross-subjects concepts have an equal prominence for all subjects. The worst result, was seen in the concepts that had a small number of training samples which do not have sufficient visual difference. On the other hand, some high classification results should be doubted, there is a possibility of small variance in comparison with a training data. Some concepts need more features to be distinguishable. In general a big consideration yielded by evaluation was a big importance of dimensions of the training set and it's quality as well.

Also, the system adds a level of semantic interpretation and understanding. This improves a connection between classification results and user queries. Besides keywords or semantic entities querying, the system offers a possibility of adding a visual concept refinement. As future work, authors focus on adding more visual features as well as feature fusion techniques in order to improve classification accuracy.

2.1.3 INHA Video Annotation Tool

The INHA-VAT video annotation system is defined as a user-friendly tool to annotate videos efficiently and accurately [40]. The main goal of the system is to turn possible a generation of a large amount of ground truth data. As seen before, the training data set is crucial for classification phases. It has a tremendous influence in the accuracy and precision of the classifier. The lack of qualified ground truth data may conduct to very poor classification results.

Manual annotation is very tedious, labor intensive and time consuming [106], [92], [64]. Hence there is a need of an automatic process. An automatic process is not always secure and may produce wrong results. Depending on the context, different requirements are needed. Thus there is a need to find a trade off between quality and time. INHA-VAT offers a semi-automatic annotation tool that support automatic annotation of the video and human annotators checking and validation.

The system is divided into various modules. The workflow begins with input of the standard video information like: camera info, recording purpose, location, date and so in *Camera and Video Information Management Module*. Objects and their attributes to be annotated stay in *Object and Attribute Management and Dynamic DB Schema Management Modules*. The last one manages a Total Annotation DB which represents the dynamic changeability of the annotation objects and their attributes. It can store various objects and their related attributes and offers an efficient search for specific frame, place and other attributes. The system support external object detection modules. These modules generate the initial annotation and should be registered in *Automatic Object Detector Registration Module* accordingly to pre-defined interface. Then, a human annotator should verify the automatically annotated data. *Manual Annotation Edit Module* is designed for it. Also the annotator can modify and add some annotations. The module offers specific tools to improve user productivity, like copying annotation data, moving ROI (Region of Interest) defining annotated objects and others. At the end, user can generate the ground truth data in XML format using a *Ground Truth Data Generation Module*. Figure 2.5 depicts an architecture of the system.

The system offers the best of two approaches: manual and automatic annotation. An automatic annotation module could be substituted or updated at any time. The user has friendly tools to manually verify and correct results as well as add new ones.

2.1.4 ViewProcFlow Annotation Tool

The system was developed to simplify and accelerate TV media content production. The overall process of production from zero is not cheap and time consuming. Thus existent material should be reused. With a large archive it is quite impossible to find the exact material that we need and here video annotation and retrieval systems solve the problem. The presented tool, called ViewProcFlow, automatically extracts metadata from video and offers tools for specific search and retrieval. The main novelty of the system is an

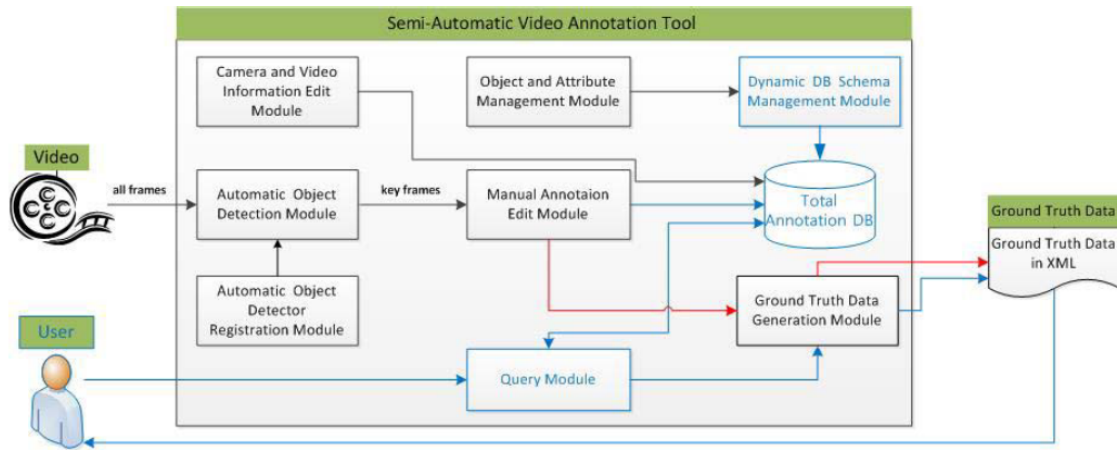


Figure 2.5: INHA-VAT Architecture

introduction of the sound recognition into annotation process [64], [65].

The ViewProcFlow video annotation system was developed in cooperation with Du-video, a multimedia content production company. Thus the authors had a direct contact with the video production professionals as well as the real data what provides a good understanding of the problem. The system is divided into a Server-side application and a Web Client-side interface. The global architecture is shown in the Figure 2.6.

Client

The client-side application is used to search, browse, visualize the video archive and metadata associated with it and includes mechanisms to validate the extracted content. Users can search and filter the obtained results. Search could be done by text or image. To improve search results, the user can specify the precision and desired area in the image search. The system offers the possibility of creating new classifiers. The user defines a new classifier by uploading new images or selecting existing ones. Since it is a web-application it is accessible from an internet browser with an internet connection.

Server

The main logic behind the server is supported by the openFrameworks library. The server waits for new videos to be processed and when the process starts it creates the XML-file with metadata related to the video. The metadata file contains information: *title*, *status*, *date*, *duration*, *description*, *config* and *Thesaurus*. *Status* reflects if the video was validated by the user or not. *Config* defines a threshold used in segmentation and the number of frames jumped after a cut. *Thesaurus* contains EUROVOC² identifiers of detected concepts. Metadata extraction process relies on separated components witch extract related features.

²EuroVoc is a multilingual, multidisciplinary thesaurus covering the activities of the EU, the European Parliament in particular. It contains terms in 23 EU languages.

The segmentation, a process of division of the video into segments using a specific condition, is done with a simple difference of histograms [19]. Basically it is an absolute difference of the sum of each pixel from the extracted frames. If it is higher than a threshold we have a new cut what indicates a new key-frame.

Face detection is done with the algorithm presented in [37]. With previously trained classifier algorithms it allows objects on the image to be detected in a real time. It produces good results but misses occluded faces.

The system supports query-by-example queries. The mechanism behind it is characterized with a visual features, an image key-points that are invariant to scale and rotation with computed descriptor. This descriptor is used for matching purposes between images. The used technique was SURF [9] and SIFT [57] available in the OpenCV framework.

Then concept detection relies on two types of features: visual and audio. User can choose one of the two. Visual features are represented by Marginal HSV Color Moments and features obtained by applying a bank of Gabor Filters along with a Regularized Least Squares classifier [36]. Audio features are characterized by spectrograms and non-negative matrix factorization (NMF) which feeds a k -NN classifier. Each sound sample is cut into 0.2 seconds intervals. Longer segments will overlap. If the concept is detected, its id is added to the XML file that describes a scene.

At the end of the development was concluded that visual information provides better results than audio. This is due to the lack of training samples for audio classifiers. Future work predicts an investigation for fusion of the features.

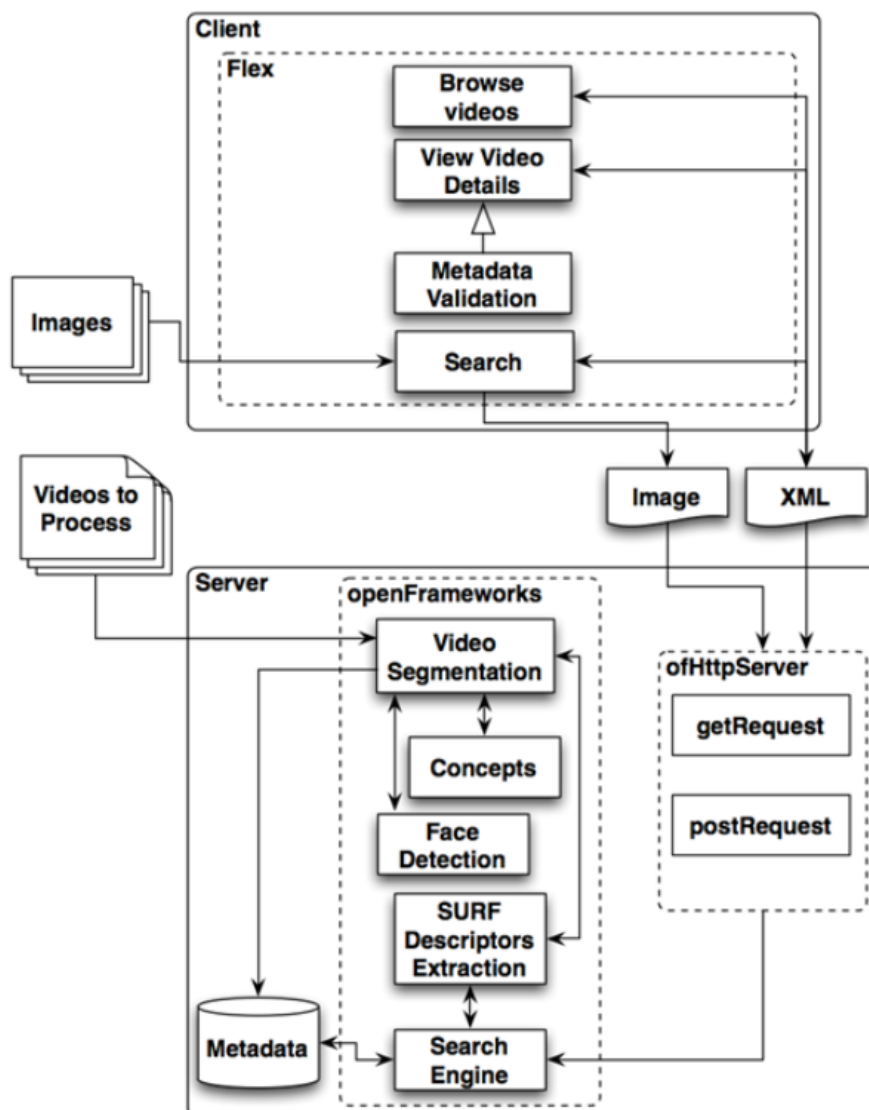


Figure 2.6: VideoProcFlow Architecture

2.2 Scene Text Detection and Recognition

Text in videos is a rich and very important source of information for content-based retrieval, since it is usually related with the video content. A majority of text extraction approaches are designed for image documents but since video is a sequences of the images, methods could be adopted. As a result text extraction process from video documents presents more challenges over images such as low resolution, low contrast and distortion [119]. The text could be separated into two groups: *Caption text* and *Scene text* [86]. Generally the first one is manually added into the scene overlaying it (e.g. subtitles, sports scores, etc). *Scene text* naturally exists in the image and could be seen in different conditions, orientations, sizes and fonts (e.g bill boards, sign boards on roads, addresses, businesses, etc). Since caption text is edited manually, it is easy to process and detect it while scene text is unpredictable, hence scene text detection is much more complex and challenging [89].

Therefore text recognition becomes a good font of information for automatic video annotations. In combination with its temporal information it could offer even more event-related information. Since the context of this thesis is related to the video, we will focus on methods designed for extraction of the scene text.

D. S. Guru et al. [27] broadly classify text detection techniques in three categories: 1) Connected Component, 2) Gradient and Edge based and 3) Texture based, but many other scientists classify just into two: Region based, same as Texture based, Sliding window, and Connected Component [105], [87], [112], [114]. *Connected Component* methods often use color and region features to localize text components. Then these components are aggregated into candidate text regions for further processing. Region selection normally is accompanied with heuristics in order to eliminate false positives. These approaches have a weak performance in frames with small fonts and cluttered backgrounds due to its assumptions that text characters should have the same intensity values and geometrical properties and text color bleeding. Recently the Maximally stable extremal regions - MSER method was developed and promise a good performance, but the problem of producing many false positives requires more study. The extraction phase is followed by grouping character candidates into text lines/paragraphs, additional filtering may be performed on top of groups. *Gradient and Edge based* has better results in detection but produces more false positives. These approaches are based on the observation that gradients of text pixels are higher than gradients of non-text pixels. They are fast, efficient and accurate but constant and adaptive thresholds create problems of false positives specially in complex backgrounds. *Region based* methods attempt to extract features in sliding window and classify the multi-scale local region as text or not text utilizing classifiers. Fast Fourier Transform and Wavelet decomposition normally are used as features passed into the classifier. These techniques work by defining appearance of text as special textures to solve the problem of complex background, but require high dimensional features and

classifiers. The image has to be processed in multiple scales. They have very high computation loads. In sum they are good in false positive elimination, but expect higher contrast for text than its background.

2.2.1 Text Localization Techniques

In this section we will describe three types of text localization techniques.

2.2.1.1 Connected Component

C. Shi et al. proposed scene text detection approach using graph model built upon Maximally Stable Extremal Regions (MSERs) [87]. This model was introduced in 2002 by J. Matas et al. It is a new type of image elements useful in wide-baseline matching: Maximally Stable Extremal Regions. These regions are defined solely by an extremal property of the intensity function in the region and on its outer boundary [63]. Detection of MSER is also related to thresholding and every extremal region is a connected component of a thresholded image. Maximally Stable Extremal Regions are the core of the proposed approach of C. Shi. Due to its robustness against view point, scale and lighting changes as well as text properties it was a natural selection. Furthermore MSERs computation is quite efficient: $\mathcal{O}(\log n \log n)$. The flowchart of the algorithm is illustrated in Figure 2.7.

At the start two kinds of MSERs are detected: light areas on dark and the dark areas on light background. To eliminate their mutual influence they are handled separately. At this stage, regions with too big or too small size are discarded as non-text. Besides this filtering, many false positive persist. Thus a next phase of labeling MSERs as text or non-text areas is employed.

As mentioned before, scene text presents a high degree of intraclass variation due to illumination conditions, fonts and distortions. Since classifiers approach is limited by a training set and heuristics by thresholds, C. Shi et al. explores a way to combine as much information as possible into one framework. The proposed framework start with a construction of undirected graph $G = (V, E)$, where each MSER is a node and the neighboring nodes are those ones that satisfy the distance between components and its size criteria. Each edge is assigned a nonnegative weight as the cost of cutting the edge. The cost function calculation is based on the unary and the pairwise cost of the neighborhood. The unary cost measures the individual penalties for labeling node as foreground or background. It focuses on the texture regularity, stroke width, occupation of the bounding box and Histogram of Oriented Gradients (HOG) [20]. The pairwise cost reflects the penalties for discontinuity between neighboring nodes. The main idea behind it is to explore the distance of color and geometric characteristics. As a result the target of the labeling problem is to find a segmentation that minimizes the cost function. To solve it, the max-flow/min-cut algorithm is used to optimize the cost function and get a binary image of the text.

MSERs labeling filters non-text areas. Now text candidates should be grouped into

lines. Grouping is controlled by the rules: merge two adjacent or overlapping components if their distance, size and color are below threshold or if one area is contained in another area, keep the larger one. The process ends when no components could be merged. Final filtering is done with a weighted sum of the classification results of the sub-image scanned on the text image. Text blocks are normalized and a trained Random Forest classifier [12] with histograms of oriented gradients features [20] classifies the 24 x 24 pixels sub-images. The ICDAR 2011 training dataset was used as a ground truth [84].

The described approach was evaluated with the challenging ICDAR 2011 text localization competition dataset [84]. The dataset has 485 images containing text in a variety of colors and fonts on many different backgrounds and orientations, 255 of those are used for testing purposes. The ICDAR standard metrics was used. The output of algorithm is a set of rectangles designating bounding boxes for detected words, named *estimate*. The other set, *targets*, is a set of ground truth. Thus, the match, m_p , between two rectangles is defined as the area of intersection divided by the area of the minimum bounding box containing both rectangles. For each rectangle from the estimation set the closest match is found in the set of targets, and vice-versa. Hence, the best match $m(r; R)$ for rectangle r in a set of rectangles R is defined by: $m(r; R) = \max \{ m_p(r; r_0) \mid r_0 \in R \}$. Derived definitions from it are the following, where T and E are sets of targets and estimates:

- **Precision** = $\frac{\sum_{r_e \in E^m(r_e, T)} 1}{|E|}$
- **Recall** = $\frac{\sum_{r_t \in T^m(r_t, E)} 1}{|T|}$
- **Standard f measure** = $\frac{1}{\frac{\alpha}{Precision} + \frac{1-\alpha}{Recall}}$, α is a relative weight between *Precision* and *Recall*, equal to 0.5.

In order to evaluate the power of the proposed approach, various experiments with different combination of features were made. Results showed that even without MSERs labeling the results are quite good, ranked as the sixth place in the ICDAR 2011 competition results, which proves suitability of MSER for text detection. Besides this, the approach was compared with other ICDAR 2011 competition approaches. Table 2.1 shows the results. This approach slightly outperforms the winner. Since ICDAR competition did not report the speed of participant methods, there is no possibility to directly compare the method in terms of computational load.

Table 2.1: Performance comparison of the C.Shi et al. method (%)

Algorithm	Precision	Recall	f
• C.Shi's method	63.1	83.3	71.8
Kim's method	62.47	82.98	71.28
Yi's method	58.09	67.22	62.32
TH-TextLoc System	57.68	66.97	61.98
Neumann's method	52.54	68.93	59.63

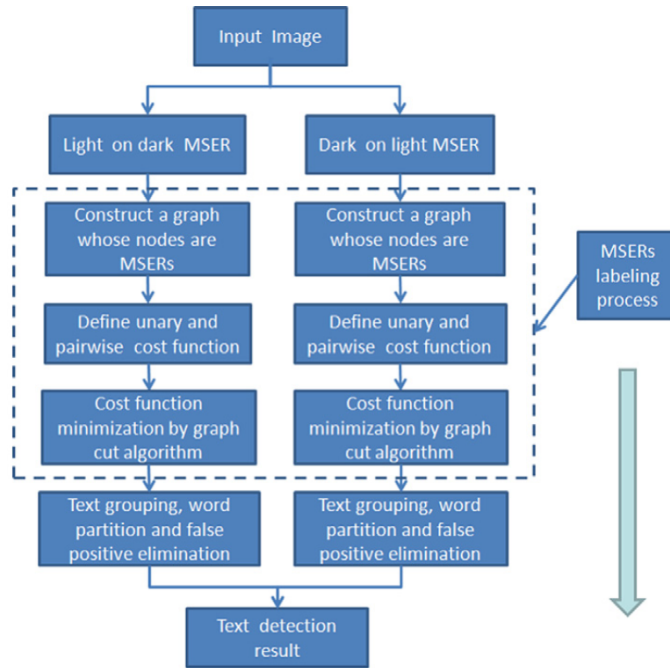


Figure 2.7: C. Shi's method workflow

In 2005, Q. Ye et al. published [113] a coarse-to-fine algorithm for text detection even under complex background. The given term, coarse-to-fine, appears due to the algorithm workflow. Firstly, it detects possible text candidates and then more sophisticated techniques analyze regions using more precise and computationally expensive features. The coarse detection phase serves to find all possible text lines in an image. It is divided in four steps:

1. Multi-scale wavelet decomposition
2. Candidate text pixels detection
3. Density-based region growing
4. Getting candidate text lines

In the first pass an image is decomposed by wavelet transformation, which gives us new representations with various depths and resolutions, as well as wavelet coefficients obtained by bandpass filtering³ which contain the intensity variety information. Candidate text pixels detection takes an advantage of the dense intensity variety around the text pixels, thus the wavelet coefficients around the pixels should have large values. Here appears a new feature, called wavelet energy, that reflects the intensity variety around a pixel in a specific level. A pixel will be a candidate text pixel if its wavelet energy feature is larger than a dynamic threshold, which is calculated manually with experiments (value proved to be the minimum value that a text pixel can be perceived by a human from

³Attenuation of very high or very low frequencies.

its background) or calculated by energy histogram to ensure that pixels from shadow areas will be detected as candidates. In the third pass, density-based region growing, text regions are formed from text pixels candidates. It defines a seed pixel (pixel with the percentage of candidate pixels in its neighborhood, 16×19 template, larger than the specific threshold), defines a label for it and collects unlabeled candidate pixels that are density-connected with a seed until the seed pixels are over. Then it labels each region as a text region and merges the other pixels with the background. To get text with different orientations, the template used in the neighborhood phase is rotated by 30 degrees six times for each seed pixel. This is the last pass of the coarse phase. Getting candidate text lines, separates text regions into text lines, using projection profile operation (vector of the sums of the pixel intensities over each column/row) [54]. Candidates with height smaller than 8 pixels or width/height < 1.0 are discarded as context.

Many textures with abrupt intensity variation, such as window curtains, leaves, could be classified as text regions resulting in false positives. Thus, a fine detection phase was developed to identify true text. This phase is composed by four passes: 1) Feature extraction in suitable scale, 2) Feature selection, 3) Training and classification and 4) Multi-scale (orientation) merging.

In the proposed algorithm, text is represented by a combination of four kinds of features, three of which are extracted in the wavelet domain and one in the gradient image. In the first pass wavelet features are extracted at the suitable scale (in the level where the candidate text lines are located) and gradients from original image. 225 features of different types could be extracted from the image. These features take advantage of frequencies behavior at various scales, intensities variances and spatial grey values distributions, histograms, correlations among adjacent pixels, crossing count histograms that reflects the distribution of the crossing counts of all scan lines and then coarsely reflect the periodicity of gradient projection maps. Feature selection serves to define a small set of the most powerful and relevant features that will be extracted. This is a very important pass because of the fact that a large set of features will decrease the generality of a classifier and increase the time of extraction and classification. Testing showed that the best number of features is 41. The third pass, training and classification, uses a support vector machine due to its easiness, smaller training samples and better generalization ability. Multi-scale (orientation) merging merges text line that were detected in more than one orientation or scale. Selections are made by rating given from classification pass and properties of width overlapping. Figure 2.8 shows the workflow of the proposed algorithm.

Q. Ye et al. [113] use two test sets for experiments. One is their own and other is from Microsoft. The last one contains 44 images with a 400×328 resolution. The test sets consist of a variety of text samples in different fonts, sizes, colors, directions, languages and quality.

The proposed method performs robustly on the majority of the test images but also has false positives. Failures were seen in cases like single character, non-rectangular text, small fonts and human-made textures that are similar to text. To evaluate the proposed

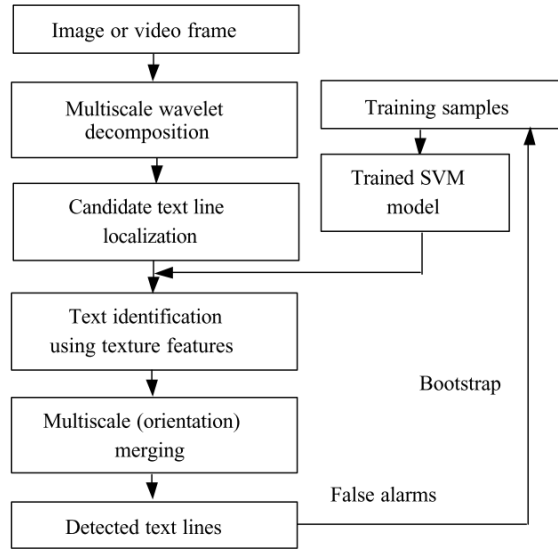


Figure 2.8: Flow chart of the [113] method

method some metrics are calculated, namely:

- **Recall rate** - Number of correctly detected text / Number of text.
- **False alarm rate** - Number of falsely detected text / Number of detected text
- **Speed** - images / second

Results can be seen in the table below.

Table 2.2: Performance comparison of Q.Ye et al. 2005 method

Algorithm	Recall rate (%)	False alarm rate (%)	Speed (images/s)
• [113] Algorithm	94.2	2.4	8.3
[53] Algorithm	91.4	5.6	1.5
[54] Algorithm	94.3	8.1	2.2

In 2014, Q.Ye & D. Doermann proposed a similar approach to detect and localize scene text. Like a previous one, it uses MSER based component extraction. In each channel (luminance, chrominance and gradient), the MSER algorithm yields components that are either darker or brighter than their surroundings. In order to get better contrast of the small text, a Gamma correction of $\gamma = 1.0$ was previously applied.

Text detection in the natural scenes requires an effective text discrimination from background and other objects. Thus, in this approach authors use a sequence of classifiers corresponding to the components. State-of-the-art HOG features are employed. Training samples are divided into smaller sizes and groups as shown in Figure 2.9. Then the gradient orientation and HOG features are calculated. In total 1296-dimensional HOG features

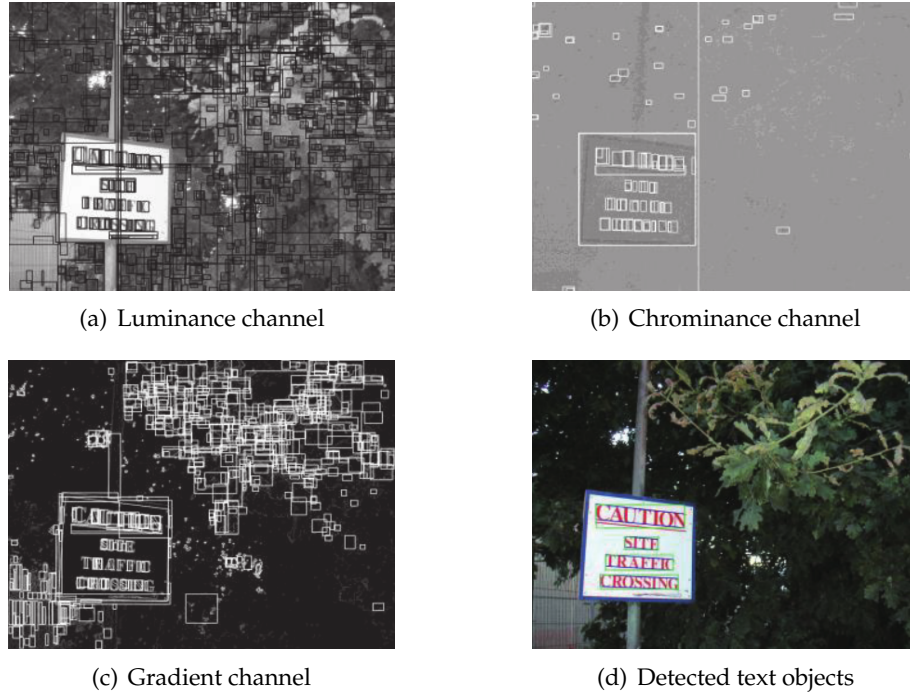


Figure 2.9: MSER bounding boxes from three channels

vector represents each sample. After this, samples are partitioned with a K -means clustering algorithm in the HOG feature space. Samples that have an aspect ratio larger than 5.0 are discarded. Remaining samples along with negative group consequently are used to train the dictionary classifier that contains K linear SVMs for the multi-class training. An one-against-all strategy is adopted. Since text could be divided into various segments, pairwise relations and holistic variance are analyzed to check isolated components. This analysis, component consensus feature extraction, includes color differences, spatial distances and alignment features. If a component is below previously defined thresholds, it is merged with the corresponding text. At the end, it yields the variance of color mean values of components. Merging two of described below processes, we obtain an integrated discriminative model. This model is based on the formula 2.1.

$$F(\tilde{X}) = F(X \cup \tilde{x}) = W^t \cdot \begin{pmatrix} \psi(X) \\ \phi(X, \tilde{x}) \end{pmatrix} - B \quad (2.1)$$

$\psi(X)$ denotes a response from the dictionary classifier and $\phi(X, \tilde{x})$ is the component consensus extraction response. W^t is a weight vector related to importance of each dimension and B is a threshold for text classification. The \tilde{x} is a component being considered for inclusion into X . Finally the output of $F(\tilde{X})$ is a positive value meaning that component is a text or negative if not.

This model, along with features described above (dictionary classifier and component consensus) are used in the training process of the linear SVM.

Since MSER generates a big number of components (hypotheses), authors decided to filter and merge them before the actual text detection process. Filters are based on some properties from component consensus features extraction, namely spatial distances and vertical overlap. Then the algorithm selects a random text hypothesis and tries to extend and test if it is a text (uses model formula 2.1). When no hypothesis can be extended, text objects that overlap each other are merged and bounding boxes are yielded. The general work flow diagram is shown in Figure 2.10.

Evaluation was done with two different datasets: ICDAR 2011 [84] and Street View Text (SVT) [103]. The first one was previously described, the SVT dataset contains text objects from Google street video frames where most of the text object are captured at middle distances. The SVT dataset is not so popular as ICDAR, so we will not discuss results from it. The ICDAR 2011 comparison table is shown on the table below 2.3. The computational cost of method on the Intel Core i5 CPU was near 0.45 and 1.6 images per second depending on the range of intensities of the MSER algorithm (Δ).

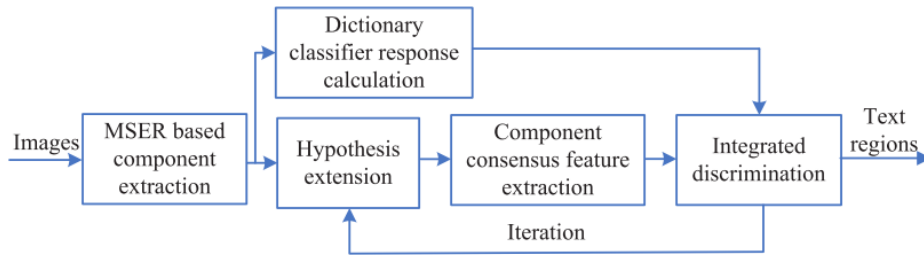


Figure 2.10: Q. Ye's method workflow

Table 2.3: Performance comparison of the Q.Ye et al. 2014 method (%)

Algorithm	Precision	Recall	f
• Q.Ye's method	64.64	83.00	72.68
C.Shi's method	63.1	83.3	71.8
Kim's method	62.47	82.98	71.28
Yi's method	58.09	67.22	62.32
TH-TextLoc System	57.68	66.97	61.98
Neumann's method	52.54	68.93	59.63

In 2012 L. Neumann et al. [69] proposed a real-time scene text localization methods, that achieved state of the art on the ICDAR 2011 dataset [85]. The real-time performance is achieved by posing the character detection problem as an efficient sequential selection from the set of Extremal Regions (ERs)⁴.

The main idea of the algorithm is that the selection of suitable ERs is done by a sequential classifier trained for character detection. It constructs a component tree of an

⁴Regions whose outer boundary pixels have strictly higher values than the region itself.

image by thresholding, increasing the value step-by-step from 0 to 255 ⁵ and linking obtained connected components from successive levels in a hierarchy by their inclusion relation. Figure 2.11 shows an example component tree.

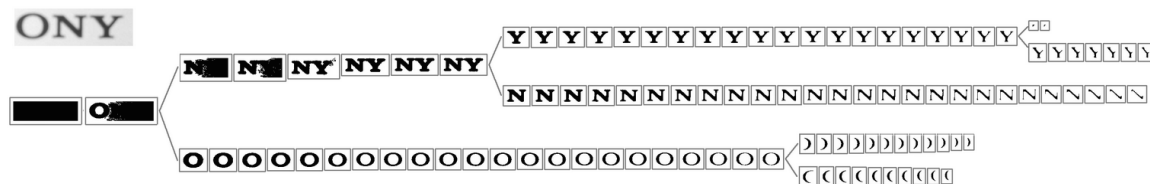


Figure 2.11: Neumann's component tree

An experimental validation showed that up to 85.6% characters are detected as ERs in a single channel and that 94.8% characters are detected within a combination of all channels, hence the algorithm performs on different channels to increase the character localization recall.

In the first stage, ERs are described by area, bounding box, perimeter, Euler number and horizontal crossings which are used to estimate the class-conditional probability P . This value is tracked using the inclusion relation of ER across all thresholds and only ERs which correspond to local maximum of the probability are selected. The classifier used at this stage is Real AdaBoost [48].

The second stage processes ERs that passed the first one. At this stage they are classified into character and non-character classes using more informative and computationally expensive features such as hole area ratio, convex hull ratio and the number of outer boundary inflection points. Classification is done with SVM with the RBF Kernel using all previously calculated features and the new ones.

After ER filtering is done and characters candidates identified, they should be grouped in high-level text blocks (words, text lines). To realize it the algorithm proposed by Lluís Gomez and Dimosthenis Karatzas [24] was used. The algorithm consists in finding meaningful groups of regions using a perceptual organization based on clustering analysis.

2.2.1.2 Gradient and Edge Based

Algorithms in this category use gradient properties of the image to detect edges. Having a transformed image that contains only edges, distances between edges are computed and searched for some type of pattern. The best known one is the similarity of distances in neighbor pixels within a horizontal or vertical orientation.

In 2010, Epstein B. et al. proposed a definition of the new image operator called SWT [22]. Stroke Width Transform transforms image from color range into stroke widths, thus every pixel contains a value of distance to the corresponding pixel on the other side of the stroke (stroke width).

⁵The algorithm could perform on the RGB or HSI color spaces.

The SWT Transformation produces an image of the same size as the original. The initial value of each element is set to ∞ and strokes are recovered by computing edges in the image using the Canny edge detector. The algorithm defines a gradient direction to each edge pixel and follows the defined ray until finding another edge pixel. The same direction is applied to the found pixel and if it is roughly opposite, each element of the path between two pixels is assigned to the width of the ray unless it already has a lower value. If an opposite pixel was not found or the direction was not opposite, the ray is discarded. Values situated in corner areas will not be true stroke widths, therefore the algorithm passes along each non-discarded ray again, computing the median SWT value and all pixels with value above median will be normalized to it.

The next step serves to find letter candidates. The proposed algorithm groups neighboring pixels using the classical Connected Component algorithm with changed association rule. In this case it is a predicate that compares the SWT values. This grouping method allows pixel with smoothly varying width to be grouped, thus regions with the text in more elaborated fonts and perspectives will be grouped. The algorithm is applied twice with different directions due to different gradients of the text (white on black or black on white). Now, with groups of connected components, the algorithm performs identification of text candidates. It computes the variance of the stroke width within each group and rejects the ones whose variance is too large. Another filters limit aspect ratio, diameter and median stroke width. At the last step, candidates that are too big or too small are discarded. Thresholds used in these steps were learned from the annotated training set, ICDAR 2003 [59]. The whole workflow of the algorithm could be seen in Figure 2.12.

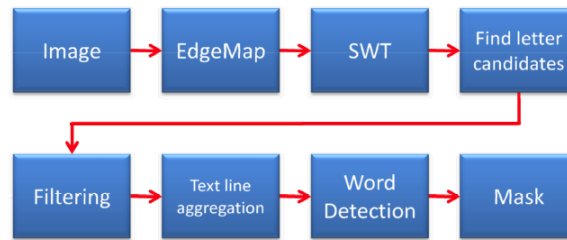


Figure 2.12: Flow chart of the [22] SWT method

To increase algorithm reliability, letters should be grouped into words. Text candidates are grouped by properties of text lines, namely stroke width, letters width and height, spaces between letters and words and their color. The process ends when no groups can be merged. Finally text is broken into separate words by heuristics of histogram based on thresholds of intra-word and inter-word letter's distances. The algorithm's performance was assessed on a ICDAR 2003 [59] and ICDAR 2005 [58] competition sets. Table 2.7 below shows results.

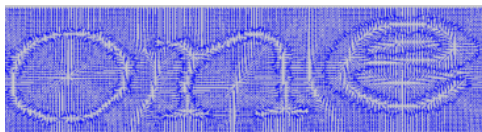
Another gradient and edge based approach was proposed by T. Phan et al. in 2012 [76]. The basis for the proposed approach is the Gradient Vector Flow (GVF) [110].

Table 2.4: Performance comparison of the SWT algorithm

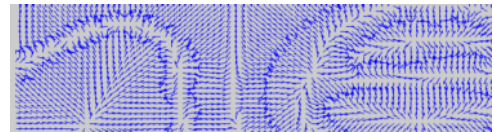
Algorithm	Precision	Recall	f	Time (sec.)
• [22] Algorithm	0.73	0.60	0.66	0.94
Hinnerk Becker	0.62	0.67	0.62	14.4
Alex Chen	0.60	0.60	0.58	0.35
Ashida	0.55	0.46	0.50	8.7
Qiang Zhu	0.33	0.40	0.33	1.6

The process starts with GVF to detect local symmetries identifying characters candidates. Then it groups characters into text lines and removes false positives with a texture analysis.

Text candidate identification takes advantage of the symmetry properties of the text, namely: *intra-character* and *inter-character*. The first symmetry is based on the inner and outer contours of the same character, while the second is between the outer contour of the left and that of the character on the right. Accordingly to these symmetries text regions are extracted with the GVF, Figure 2.13. The Gradient Vector Flow technique propagates the gradient information (the magnitude and direction) from nearby regions into homogeneous regions. To compute the GVF field algorithm uses a minimizing function with the edge map of the input image. The most important property of the GVF field is that starting from any point, we can reach a nearby edge just by following the GVF directions. Thus the regions where two neighboring GVF arrows are opposite of each other indicate a presence of local symmetry points. T. Phan proposes conditions where vectors should be opposite and the angle between should be greater than $\pi/6$. To find edges authors use the Sobel and the Canny edge map. The combination of two maps improves recall of the truly text candidates. The combination consists in the intersection of two edge maps with defined threshold, non-text symmetry components are removed.



(a) GVF field



(b) Zoomed-in GVF field

Figure 2.13: Gradient Vector Flow examples

At this stage the remaining symmetry components are horizontally grouped. In each iteration a new group is created which contains the first unassigned SC (symmetry component). The order is from top left to bottom right. If near current groups some unassigned SC exists and it satisfies similarity constraints it is added into the groups and the algorithm re-examines the expanded neighborhood. Similarity constraints compare height, stroke thickness, space and colors between character groups. The final grouping output filter small groups, so only groups with three or more SCs are retained. The

overlapping groups are merged and the union of two bounding boxes is yield.

Even though symmetry and grouping constraints, some background text-like patterns appear. Thus authors perform local texture analysis for text verification purposes. Texture analysis is done with the HOG features [20] and SVM. The training data set counts 11,600 positive and 14,100 negative samples. Each patch has a size of 48×48 and is divided into three vertical parts. The division is done because of ascender and descender text that have texture slightly different to the middle. Experiments showed that division of the patch improves false positive elimination. The classification phase normalizes each region to 48-pixel height then a window slides across. At each position SVM estimates the confidence score, then the overall score is computed by a weighted average following Gaussian distribution. If the score is non-negative, the region is considered as a text containing one.

Authors performed experiments on two public datasets: ICDAR 2013 [60] and Microsoft Text Detection dataset (MS) [22]. The proposed approach was able to identify text even with the stylish fonts, blurring, partial occlusion and complex background. On the ICDAR 2005 (dataset from 2003) competition it outperformed an existing state-of-the-art methods in terms of f -measure. In addition, experiments were done with and without HOG-features and proved that the use of it improve the precision. Table below shows these results.

Table 2.5: Performance comparison of the GVF algorithm

Algorithm	Precision	Recall	f
• [76] Algorithm with HOG	0.70	0.69	0.69
• [76] Algorithm without HOG	0.63	0.69	0.66
[22] Algorithm	0.73	0.60	0.66
Alex Chen	0.60	0.60	0.58

As a conclusion and future work, technique shows a good performance and possibility of use in general for objects, contour reconstruction and shape matching.

2.2.1.3 Texture Based

Texture based methods treat text as a special texture. They apply various transforms (such as Fast Fourier, DCT, Wavelet or Gabor filters) and classify text relying on its properties extracted from these transformations.

P. Shivakumara et al. [90] proposed robust wavelet, statistical features and central moments based method with k-means clustering for detecting text in video images.

The method applies a single level 2D Haar wavelet decomposition [62] and three high frequency sub-bands images reconstructed by inverse 2D wavelet transform. Experiments on Haar wavelet basis proved that it has good ability to characterize the text and are computationally efficient [62]. After decomposing wavelets, the algorithm performs calculations of desired features, namely energy, entropy, inertia, local homogeneity,

mean, second-order and third-order central moments. These calculations are made on a sliding window for each sub-band image. Each sub-band image has 7 features and since authors use 3 sub-bands, we have 21 features extracted at all, that are normalized into the range from 0 to 1 and passed into feature vector for every pixel in the frame.

At the final phase, classification, the K-means algorithm is applied. It classifies previously obtained feature vectors into two clusters: text and background candidates. The areas of both are calculated and the one with the smallest is defined to be the text cluster. To discard small objects, morphological operations (opening and dilatation) are performed. Then average image AV is calculated from horizontal, vertical, diagonal sub-band images and images from a morphological result. At the end of this phase we have text blocks detected in the form of boundary boxes.

Obtained results may contain some false positives, and that is why the algorithm performs some heuristics, based on height, width, and area of the text blocks. Some texture properties are used in the proposed heuristics too: Sobel and Canny edge components. Text blocks that do satisfy any of the proposed heuristics are marked as false positives, hence are eliminated.

To experiment with this method, the authors created their own set of variety of video images taken from movies, news clips and others. The total set was constructed by 101 images with 491 actual text blocks.

The experimental results of [90], show that the proposed method was better leader throw the competition in terms of detection rate, false positive rate and missed detection rate. Experiments were made even on non-text images, where the algorithm returned 32 false positives on 28 images out of 100. The metrics proposed by the paper were:

- **DR** - Detection Rate - Number of detected blocks that contains text fully or partially, truly detected (**TDB**) / Number of Actual Text Blocks (**ATB**) (manually calculated).
- **FPR** - False positive rate - Number of detected blocks that does not contain text, falsely detected (**FDB**) / Number of (**TDB** + **FDB**)
- **MDR** - Misdetection rate - Number of truly detected text blocks that miss some characters, (**MDB**)/ Number of **TDB**.

Results could be seen on the table below.

Table 2.6: Performance comparison of P. Shivakumara et al. method in (%)

Method	DR	FPR	MDR
• [90] Algorithm	96.7	4.2	5.6
[56] Edge based Algorithm	80.0	18.3	20.1
[109] Gradient based Algorithm	71.0	12.0	10.0

In 2014 the same author, P. Shivakumara et al. proposed a new approach for text detection [89]. Since this approach is designed for the video, it is divided into two phases:

search for the text candidates and multi-oriented text detection. The first one is designed to filter frames in order to select those who possibly contain text, to reduce unnecessary computational cost. Then multi-orientation provides more depth analyzes to find text areas.

Text frame classification resizes frames into 256x256 to facilitate implementation. Then the frame is decomposed with Wavelet (Haar) decomposition. Similarly to the previous P. Shivakumara's approach decomposition provides high contrast values for text pixels in high frequency sub-bands. The sample is divided into small, no overlapping blocks. Then for each block median moments are computed. These moments with respect to the mean for wavelet decomposition help in discrimination of text and non-text pixels. Discrimination is done with the K-means algorithm with $k = 2$, to cluster feature vectors of the blocks. At the end the algorithm gives a set of text and a set of non-text blocks. Even with some misclassifications *Probable Text Block Classification* significantly reduces future computational cost, as example an image from 16 blocks was reduced to 5 probable text blocks.

Following frame decomposition and candidates generation, each block is divided again, now in 4x4 pixel windows. For each window a 2nd order an 3rd order median moments are calculated as well as features extracted and normalized. Now a K-means cluster with $k = 2$ is applied to cluster 16 given features vectors to classify pixels into text and non-text. As before, higher values mean presence of the text. In the next step small blobs and isolated straight components are eliminated. After this, the Percentage of Pixel Based Symmetry Property is introduced. As the name indicates it is a symmetry property based on the observation of the text pixel region distribution over the four quadrants. The block is divided with respect to x and y axes with the origin in the center of the block. The method counts text pixels, yielding a percentage for each quadrant. Quadrant with no text pixels indicates block without text, thus it is immediately discarded. Symmetry is calculated with clustering of the values of the quadrants. The frame is classified as a text frame if at least one block satisfies proposed above symmetry, otherwise it is a non-text frame.

At this point we have text candidates and a frame with them, but candidates may not be the complete information of text in the frame. Hence for each text candidate authors get the corresponding edge components in the Sobel edge map. If the centroid of the representative does not fall into its major axis, the candidate is eliminated as a false positive. Subsequently filtered candidates are used for fixing bounding boxes which are used in *Angle projection boundary growing*. This phase was developed to determine the angle of orientation of a text line from the candidate text representatives. Each candidate text representative is considered as a seed point for growing. Boundary grows from seed pixel by pixel until it gets a white pixel of the nearest neighbor component in the text line. The process continues along the text edge information in the Sobel edge map. The stop condition in this case is the satisfaction of some convergent criteria. When the boundary growing is finished, the angle is computed using PCA. Authors use PCA supposing that

is better in distinguishing form objects that appear isolated. The computed angle for the first edge component is considered as the first iteration angle. The boundary is grown until it reaches the nearest edge component and then two components are merged to make one boundary. The angle of the merged components is computed with PCA too and is considered as the second iteration angle. If the absolute difference between two iterations is less than 1° , the angle calculation stops and the last iteration angle is used as the projection angle to draw the top and bottom boundaries for the text line. In other cases, the iteration continues. If the angle computation stops upon convergence, the boundary growing process continues, this is the advantage and a novelty of the proposed method. However, the first three angles are not checked with the convergence criteria because they may not be the actual text direction due to insufficient edge information.

Even with many filters made above, some false positives appear. Thus the authors propose the final false positive elimination based on two heuristics. The first one is based on the distances between centroids of edge components in a text line. The second one is based on the angle of the text line.

Experiments realized in this work showed that the proposed method achieves the best accuracy for ICDAR 2003 [60] data according to its official measures (Recall: 0.62; Precision: 0.74; f : 0.68). Besides this they showed a good connection and workflow between various components of the approach. However the method fails to detect perspective distorted complete text lines. Future work addresses this problem by exploring temporal information as well as curved text lines using background and foreground information.

2.2.2 Optical Character Recognition

We already discussed the importance of text recognition in the scene images. This action could be decomposed in two steps: detection and recognition. The first one was discussed above, the second will be discussed in this section.

In classical OCR problems text is presented by monotone characters on fixed background. Algorithms developed for this type of environment achieve very high results. Scene text is much more complicated to recognize, because of its background variations, including lighting, texture and font. As solution to this problem feature learning algorithms were applied. They are appropriate for this type of problems but some problems continue to be unsolved. Besides their disadvantage in the computational factor, they can generate an excessive number of features.

Researching in the area of text recognition considered various solutions. Some of them focus on the simple classifiers trained on hand-coded features, some on the multistage pipelines. The most common features used in classification include gradients, edges, texture or shape [70]. Besides standard approximations, neural-networks architectures have been employed too, specially for the written text. Additionally tree-structured approaches build a tree minimizing the cost-function [67, 88]. Generally text recognition

approaches can be divided into two groups based on how do they deal with the problem. The first one exploits a sliding window that localizes individual character or even whole words. The advantages of these approaches are the robustness to noise and blur due to their properties of exploring features aggregated over the whole ROI⁶. But it can produce a big number of rectangles that needs to be evaluated. The second localizes individual characters as connected components using local properties of an image. Thus a character localization and segmentation of all scales and orientations can be detected in one pass and passed to an OCR stage. The drawback of these methods is the sensibility to small changes like noise which cause an unproportional change in the selected region decreasing classification as well [70].

Coates A. et al. proposed an algorithm to solve a problem of text recognition [16]. They designed a very interesting architecture that consists in the application of an unsupervised learning algorithm that will generate the features used for classification. The basic idea comes from a convolutional neural network [47], where authors presented a single network which learns the entire recognition operation, going from the normalized image of the character to the final classification. Presented training method can be used to rapidly construct extremely large sets of features with minimal tuning. Then the number of feature is reduced with spatial pooling and a linear classifier is trained for text detection or recognition.

1 - Feature learning

This is the key component because it is the basis of the whole system. The main idea is to employ a unsupervised learning algorithm to generate features. Authors use a variant of K-means clustering because of its simpleness and speed. They collect a set of small grayscale text image patches from a training dataset, prepare it with statistical pre-processing and finally run a an algorithm to build a mapping from input patches to a feature vector.

2 - Feature extraction

The proposed classifier considers 32-by-32 pixel images. Previously builded features vectors were made for every 8-by-8 sub-patch input, yielding a 25-by-25 representation. The dimensionality of representation is reduced with an average pooling where they combine the responses of a feature at multiple locations into a single feature. The result to this phase produces a vector of $9d$ features.

3 - Text detector training

Trains a binary classifier that aims to distinguish 32-by-32 windows that contain text from windows that do not. To train the classifier authors used the ICDAR 2003 training dataset. Text areas are collected then it extracts above described features features and feed a linear SVM.

⁶Region of interest

4 - Character classifier training

For character classification, the authors also use a fixed-sized input image of 32-by-32 pixels, which is applied to the character images in a labeled and cropped dataset. Since feature production could produce a large number of features, a small training dataset may be a cause of the over-fitting problem. To solve it, authors combine different datasets into the learning phase. Moreover, they also experimented with synthetic augmentations of these datasets. These synthetic generated examples include copies of samples with random distortions

The table below shows results of experiments obtained on the ICDAR 2003 dataset [59].

Table 2.7: Test recognition accuracy on ICDAR 2003 character sets in (%)

Algorithm	Test-62	Sample-62	Sample-36
• [16] Algorithm	81.7	81.4	85.5
[68] Algorithm	67.0	-	-
[115] Algorithm	-	81.4	-
[81] Algorithm	-	-	84.5

In 2013 C. Shi et al. proposed a scene text recognition using part-based tree-structured character detection [88]. First the authors detect characters with a part-based tree-structured models, then they build CRF⁷ model on the potential locations. After this, a word recognition is obtained with a minimizing cost function.

The base of text detection is the modeling of the character as a tree whose nodes correspond to parts of the character. In this way, global information and the local appearance are incorporated into the tree-structured model. The model is composed with local appearance model which reflect the suitability of putting the part based models on the corresponding position along with the HOG local appearance descriptor. Another component is the global structure model which scores the character-specific global structure arrangement of configuration. Basically, it combines different image parts and evaluates it with a dynamic programming and maximizing function. To improve detection, model is re-scored considering the similar intensities of the different parts.

For each type of character, the authors construct a tree-structured model. Learning is proceeded as a fully-supervised paradigm with provided positive images with characters and part labels and negative - without characters. Thus the model is discriminatively learned using a structured prediction framework.

The character detection step provides a set of windows with high confidence, but some false positives and ambiguities arise. Thus, authors use a language model and spatial constraints to eliminate these concerns. Hence C. Shi builds a CRF model on these detection windows. Previously obtained information along linguistic knowledge defines a cost function. As a result, recognition requires a minimizing cost function calculation.

⁷Conditional Random Field

At the start of recognition step, a Non-Maximum Suppression is applied on the original character detection results [121]. Then, for each location authors search for potential candidate characters. Candidates are chosen from detection windows that are close to the selected location. Thus a new graph is constructed. Then the constructed graph is evaluated with a cost function. The cost function is composed with the unary cost and pairwise costs. The first one represents the penalty of assigning label to node and reflects the confidence for the class character label. The second, pairwise, incorporates linguistic knowledge and spatial constraints. Authors use the SRI Language Toolkit [94] to learn the probability of joint occurrences of character in a large English dictionary⁸.

Finally, after computing the unary and pairwise costs, authors use the sequential tree-re-weighted message passing (TRW-S) algorithm to minimize the cost function [41].

The evaluation of the performance of the proposed method was done on the four public datasets: Chars74k [21], ICDAR 2003 [59] robust character recognition, Street View Text (SVT) [104] and ICDAR 2011 word recognition [85]. Since other approaches described here did not use Chars74k dataset for testing, we will focus only on ICDAR. For all datasets the same bigram language model learnt from the lexicon with 0.5 million words is used. Furthermore, for ICDAR datasets, authors measure performance using a lexicon created from all the words in the test (ICDAR FULL) and with a lexicon from a training set plus 50 random words from the test set (ICDAR 50). The results are presented in a table below 2.8.

Table 2.8: Word recognition rates of C. Shi et al. method

Method	ICDAR03(FULL)	ICDAR03(50)	ICDAR11(FULL)	ICDAR11(50)
• [88] Method	79.30	87.44	82.87	87.04
[67] Method	67.79	81.78	-	-
[66] Method	-	81.78	-	-
[1] Method	55	56	-	-

Apart of scientific proposals there are some web services to perform OCR.

Google Docs [25]

Google docs offers an optical character recognition service that converts files with typewritten/printed text into editable documents. The service will attempt to extract the text from the PDFs or images, creating a new Google Doc for each image. Supported file formats are *pdf*, *jpeg*, *png*, *gif*, and for good accuracy it is desirable a 10 pixel character's height. The OCR works only for the latin character sets and due to computational load can take up to 30 seconds.

Abbyy FineReader [2]

Abbyy is a company that develops and delivers to the market a wide range of high-tech products and services that are based on document recognition and linguistic

⁸Dictionary with around 0.5 million words provided by C. Shi et al.

technologies. One of their products is a cloud OCR which provides the SDK in the form of Web API that enables communication with a remote server via HTTP. It is possible to load images to the OCR server, process with necessary recognition and export parameters, and obtain the results of processing. The service supports many file formats for input (*bmp, png, jpg, pdf, gif, etc.*) and for output (*txt, rtf, xml, etc.*) and various languages (including Chinese, Arabic and other non-latin languages). Abbyy offers an extended range of plans and pricing.

OnlineOCR.net [72]

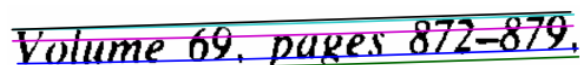
OnlineOCR.net offers a SOAP Web service, which performs OCR on an image and returns the resulting document or recognized text. It supports 28 recognition languages, different input and output formats (*jpeg, gif, Text Plain, rtf, pdf, etc.*) and offers different pricing according to the number of recognition images.

All services described above have an excellent recognition accuracy for the text from documents. The last one better preserves formatting, but this is not relevant in the current project.

Another very used OCR engine is Tesseract. Tesseract is an open-source Optical Character Recognition engine that was born at HP laboratories between 1984 and 1994. Then, in late 2005, HP released it for open source. Nowadays it is extensively improved by Google.

The engine starts with a connected component analysis in which outlines of the components are stored. Extracted outlines are gathered together into *Blobs*. After this, blobs are organized into text lines. Yielded lines and regions are analyzed for fixed pitch or proportional text. The algorithm analyses a spacing between character and accordingly to different spacing breaks into words. Then the recognition phase begins. It is a two-pass process. Obtained words from the previous step are being recognized and passed into an adaptive classifier as a training data. Thus the classifier get a chance to more accurately recognize text lower down the page. A second pass analyzes a page for a second time to recognize missed words. A final phase resolves fuzzy spaces and checks alternative hypothesis to locate small-cap text. The text below deeply explains how does tesseract engine works.

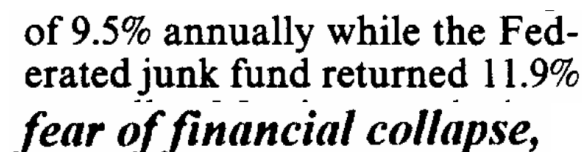
Since text pages are provided by the user, it could be a skewed page. In order to deal with it, the line finding algorithm is employed. It separates letters into blobs and at the end merges overlapping ones. The yielded text lines are fitted more precisely using a quadratic spline. Thus tesseract could handle pages with curved baselines. Figure 2.14 shows an example of the baselines overlapping text line.



Volume 69, pages 872-879.

Figure 2.14: Tesseract OCR baselines

Tesseract test the text lines if they are fixed pitch. The fixed pitch text words are chopped into character using the calculated pitch. The non-fixed-pitch (proportional) text lines like "11.9%", which have different spacing between tens and units or for example space between "of" and "financial", where no horizontal gap between bounding boxes, are differently treated by Tesseract, Figure 2.15. Generally it measures gaps in a limited vertical range and spaces that are close to the threshold are made fuzzy. Thus a final decision can be made after word recognition.



of 9.5% annually while the Fed-
erated junk fund returned 11.9%
fear of financial collapse,

Figure 2.15: Tesseract spacing example

The word recognition starts with the classification of the words with the initial segmentation provided from line finding step. The rest of the steps applies to non-fixed-pitch text. While the recognition result of the word is unsatisfactory, Tesseract chops the blob with the worst confidence from the character classifier. If the confidence is still low, the word is passed into associator. The associator makes an A* (best first) search of the segmentation graph of possible combinations of the maximally chopped blobs into candidate characters.

The static character classifier started with the topological features, involved polygonal approximation but it was insufficient. Thus the breakthrough idea was that the features in the unknown need not be the same as the features in the training data.

The classification phase is divided into two-step processes. In the first step, a *class pruner* creates a shortlist of characters classes that the unknown might match. A coarsely quantized 3-dimensional look-up table is used to fetch feature a bit-vector of classes. The bit-vectors are summed over all the features. After correcting for expected number of features, the classes with the highest counts become the short-list for the next step. Then in the second step the feature of the unknown looks up a bit vector of prototypes of the given class that it might match. Each prototype is represented by a *configuration*, which is a logical sum-of-product expression with each term. Thus the distance calculation process keeps a record of the total similarity evidence of each feature in each configuration of each prototype. At the the end the best combined distance over the stored configurations is calculated.

As expected, an OCR engine should contain a linguistic analysis. For each new segmentation Tesseract chooses the best available word string in each pre-defined categories. The final decision for a given segmentation is the lowest total distance rating multiplied by respected constant of the category. Different segmentation produce different numbers of characters in the words, because of this it is hard to compare words directly. The solution used in Tesseract is a generation of two numbers for each character classification.

The first one is a confidence (minus the normalized distance from the prototype). The second is a rating, which multiplies the normalized distance from the prototype by the total outline length in the unknown character.

Apart from the static classifier, Tesseract employs an adaptive classifier too. The last one uses isotropic baseline/x-height normalization, which makes easier to distinguish upper and lower case characters and improves noise filtering.

Currently Tesseract is the leading commercial engine in terms of accuracy and is widely used in computer vision applications.

2.3 People Identification and Recognition

The human face is easily detected and recognized. In contrast to other biometrics such as fingerprint or eye iris recognition, face is much more obvious and easier to obtain. This is one of the most used human pattern recognition features. Also faces are ubiquitous in photos, videos or news. As a consequence the importance of it is unquestionable. For many years face recognition was and continues to be one of the most successful application of image analysis and understanding. It received significant attention during recent years. Surveillance systems, closed circuit TV monitoring, gaming, human-machine interaction, security, information extraction and management are some examples of its use.

W. Zhao and A. Rosenfeld mention the problem of automatic face recognition and divides it into three key steps/subtasks: detection, feature extraction/normalization and identification [120]. Figure 2.17, proposed by N. Patil et al. [75] shows it. The first step is already done in the VideoFlow project [64]. Face detection is done with the Viola and Jones algorithm [37].

In general it is a machine learning the algorithm where a cascade function (classifier) is trained from positive and negative images. Each image is analyzed for Haar features extraction, Figure 2.16. Each feature is a single value obtained by subtracting the sum of pixels under the white rectangle from the sum of pixels under the black rectangle. For faster processing, input image is transformed into integral images, explained in section 2.4. Then the relevant features are calculated and selected with Adaboost algorithm, which finds the best threshold that will classify the faces to positive and negative. Furthermore, authors employ the concept of Cascade of Classifiers. It groups the features into different stages of classifiers and applies them one-by-one.

A recent survey done by Cha Zhang and Zhengyou Zhang show us that the algorithm is very simple and effective for frontal face detection, but it has some problems with faces at arbitrary poses [118]. Having the faces detected, our task becomes to complete a process with other two steps: normalization and identification.

Generally the problem of face recognition can be formulated as a process of identifying or verifying one or more persons in the scene obtained from video or images using

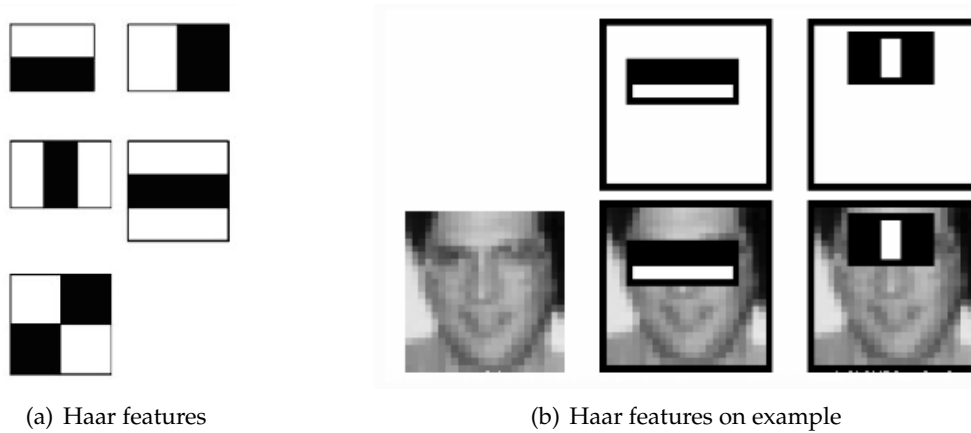


Figure 2.16: Haar features

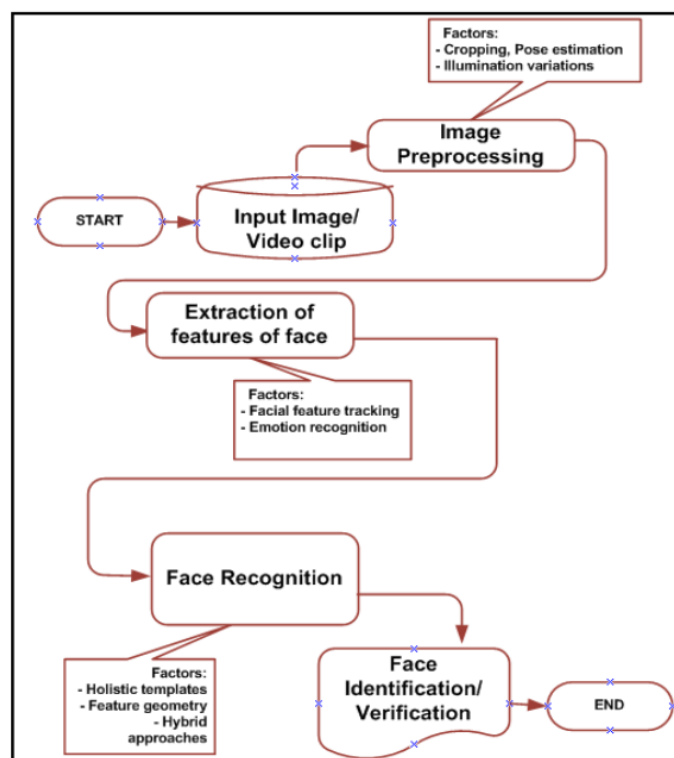


Figure 2.17: Face detection process workflow

a stored database of faces [120]. The identification process deals with the input of unknown faces providing back the determined identity from a database of known individuals. While in verification process, the system just confirms or rejects the claimed identity of the input face.

First approaches treated face recognition as a 2D pattern recognition problem. The typical pattern classification techniques were applied, which use measured attributes of features like the distances between important points. Later commercial opportunities, the availability of real-time hardware and surveillance needs provoked an increase of the interest to the area. As a result holistic approaches appeared. In contrast to the feature-based approaches, holistic approaches are more sensitive to variations in illumination, camera viewpoint and accuracy in face localization. However, both types of approaches require a good accuracy of face acquisition.

Besides the face, humans offer some collateral information such as race, age, gender, speech that may be used in narrowing the search and improving recognition as well. Psychophysics and neuroscience studies showed that additional knowledge can contextualize face recognition process, specially where they are supposed to be located. The presence of dominant features, namely the odd ones like big ears, a crooked nose influence a face recognition process done by humans. So, it is a good idea to consider these features in recognition process provided by machines.

The study of Shepherd et al. 1981 found that for the face recognition the upper part of the face is more useful than the lower part. Also a study proved that the of aesthetic attributes such as beauty and attractiveness has an influence in recognition rates. The more attractive face are better recognized, then the least attractive followed by the midrange faces. In relation to the viewpoint some experiments suggest that memory for face is highly viewpoint-dependent.

Depending on the nature of the application, the system has different requirements, hardware and outputs. As a result, different sizes of the training and testing databases, clutter and variability of the background, noise, occlusion and speed requirements become to be a challenge in the system development.

The first step is to detect a presence of the faces and their localizations. Then the feature extraction task is employed which obtains required features and feeds them into a face classification system.

Since different classification systems exist, different features may be extracted. The feature range could be very large, from the local features like: eyes, nose and mouth to the abstract ones, holistic, where the whole face is extracted. In many cases, holistic approaches use facial features in order to arrange faces into a pre-defined form (normalization process). W.Zhao et al. classifies feature extraction methods into three categories: generic, which are based on edges, lines and curves; feature-template-based that use to detect facial features such as eyes; structural matching methods that consider geometrical constants on the features. Methods with the focus on the the individual features suffer from difficulties of changing the appearances of the feature (ex: closed eyes, glasses).

Presently these methods are replaced with the recent statistical ones which are more robust. The feature matching methods could be categorized into three ones:

Holistic matching

Methods that use the whole face region as the raw input to a recognition system. The recognition rate in this type of methods is very dependent on the accuracy of face detection. The most widely used representation of the face region is eigenpictures, which are based on PCA (Principal Component Analysis).

Feature-based (structural) matching

Methods that extract local features (eyes, nose and mouth), extract their locations and local statistics like geometry/appearance and then are fed into a structural classifier.

Hybrid methods

The humans use both local features and holistic ones, so it is expectable that someone will design a machine with such type of recognition method. Perhaps these methods could potentially offer the best of the two types of methods.

As written above, the most known face representation in holistic matching methods is an eigenpicture. It is based on PCA⁹ that explores statistical redundancies in natural images. Since the face images are normalized with respect to scale, translation and rotation, the redundancy is even greater. It encodes a face image as efficiently as possible. In mathematical terms finds a principal components or the eigenvectors of the covariance matrix of the set of face images, treating an image as a point (or vector) in a very high dimensional space. PCA reconstructed images are much better than the original distorted ones. Turk and Pentland in 1991 demonstrated face recognition machine with eigenpictures (eigenfaces). Given the eigenfaces, every face in the database can be represented as a vector of weights. These weights are obtained by projecting the image into eigenface components. Every new unknown image is represented by its vector of weights, then the identification is done by location the image in the database whose weights are the closest to the weights of the test image.

Another systems that use Linear or Fisher discriminant analysis (LDA/FLD) have also been very successful in face recognition. In all these projection algorithms, classification is performed by projection the input x into a subspace via projection matrix P , then the projecting coefficient vector of the input is compared with the pre-stored projection vectors of labeled classes. At the end, the input class label is determined. The comparing process varies in different implementations. It could be the Euclidean distance (weighted or unweighted).

In the feature-based structural matching methods geometry of local features is explored. A typical local feature representation consists of wavelength coefficients for different scales and rotations based on fixed wavelet bases. These coefficients are robust to

⁹Principal Component Analysis

illumination change, translation, distortion, rotation and scaling. Normally such type of system can be characterized as graph matching methods.

The hybrid methods are the mix of holistic and feature-based methods. As an example in 1994 Pentland et al presented the modular eigenfaces approach. The capabilities of eigenfaces were extended in several directions: classical eigenfaces and view-based eigenspaces. The first one uses all images from all views to construct a eigenspace, while the second one separates them, as a result the collection of images taken from each view has its own eigenspace. Thus the second approach performed better. Another combination of eigenfaces is an extension to eigenfeatures including: eigeneyes, eigenmouth, etc. Another approaches follow the idea of hybrid PCA and LFA (Local Feature Analysis). LFA is a biologically inspired feature analysis.

The human retina has a huge array of receptors and a small fraction of them are active and correspond to natural objects that are statistically redundant. The activity of these sensors forces the brain to discover where and what objects are in the field of view and recover their attributes. The motivation for using components is that head pose changes lead to changes in the positions of facial components. However, such type of systems requires a large number of training images taken from different views and under different lighting conditions. Another drawback is that they cannot be applied to small images.

Face recognition from image sequences (video) is another field of study, as motion can improve the recognition process, but significant challenges still exist, such as:

- *The quality of video is low*, depending on the video acquisition place, process and hardware subjects may suffer from large illumination and pose variations. In addition, partial occlusion and disguise are possible.
- *Face images are small*, image resolution affect the accuracy of face segmentation along with detection of its landmarks in order to extract required features.
- *The characteristics of faces/human body parts are very wide*, the intraclass variations of human bodies, specially faces represents a big challenge for developing a generic descriptor and classifier.

To realize a full potential of video-based face recognition three closely related techniques should be explored: face segmentation/pose estimation, face tracking and face modeling.

As the name indicates, face segmentation separates a face object from the other ones, while the pose estimation calculates the location of facial features. Early attempts used simple pixel-based change detection procedures based on difference images, but the presence of moving objects and occlusions introduce difficulties. The color information may speed up the process. Given a face image, important facial features can be located and then used for pose estimation.

After a face and features location detection, detected features can be tracked. The feature tracking are critical for reconstructing a face model from a Structure from Motion

(SfM). Also it plays a crucial role in spatiotemporal-based recognition methods which directly use its information [49]. The face tracking can be divided into three categories proposed by [120]: head tracking, facial tracking and complete tracking. The first one involves the motion tracking of a rigid object that performs rotations and translations. The second one involves explores an anatomy of the head that is deformable motion. The last one as a name indicates tracks a full face along with its facial features.

The last technique is face modeling which includes a 3D shape and texture modeling. In computer vision the most used methods of estimating 3D shape from a video sequences is SfM, which estimates the 3D depths of interesting points.

The techniques discussed in the paragraphs above represent an improvement of video-based face-recognition methods over the still-image-based ones. Application of those offers a possibility of construction of a virtual frontal view that can be synthesized via pose and depth estimation from video. Another advantage of video is an abundance of frames. It can improve the recognition by the using of "voting" based on the recognition results from each frame. The drawback of the last one is the computational expense. To overpass this drawback video skimming can be applied. This technique was used by [108] to reduce the number of frames to be processed. The video break module detected key-frames based on optical-flow method along with the simple pair-wise frame differencing.

However, unpredictable poses, facial expressions and ornamentations, occlusions and illuminations conditions turn the recognition process into one of the most challenging problems. The dynamic environment of the video is a good example where these constraints are present. As a result two of the most challenging issues in face recognition are illumination and pose variation.

As Zhao et al. concludes, the face recognition evaluations revealed two major challenges: the illumination and the pose variation problem. In addition, images can contain partially occluded faces. If face images are acquired in an uncontrolled environment, these problems are unavoidable. The experiments of Adini et al in 1997 proved an importance of illumination problem: the changes introduced by it are often larger than the differences between individuals what conducts to the misclassifications of the system. To solve it many approaches were developed. Also, these approaches fall into the normalization step. Zhao divides them into four types:

- Heuristic methods (ex: discarding principal components), in general these methods use heuristic methods to compensate lighting influence. Many of them use a simple contrast normalization and histogram equalizations. Another very used method is discarding a few most significant principal components (normally three) which improves recognition rate.
- Image comparison methods (methods with appropriate image representation and distance measures), approaches based on these methods use different image representations like edge maps, derivatives of the gray levels and 2D Gabor filters.

- Class-based methods (multiple images of the same object in the same position but in different lighting conditions), the main idea is to capture face images under ideal assumptions and different lighting conditions.
- Model-based methods (3D models), 3D model is used to synthesize the virtual image.

The pose problem, is another big challenge. To solve it, researchers have proposed various methods, which can be divided into three classes:

- Multiview image methods (multiview database images of each person available), input image is aligned to database images corresponding to its pose and then template matched. The drawback of it is a need of storing many different images per person, no lighting or facial expressions variations are allowed and since iterative searching is involved, the computation cost is high. A. Lameira et al. aim to realize it without distorting the object area, thus the following steps are performed. Firstly, a quadrangular region is created and the object area is placed inside it, while the remaining space is filled with black pixels. Finally the obtained image is resampled to a default resolution [45].
- Hybrid methods (multiview training images and one during recognition are available), this type became the most successful and practically used. The good example of it is the popular eigenface approach mentioned above. The basic idea is to code the pose information by constructing an individual eigenface for each pose.
- Single-image/shape-based, this type of approaches includes low-level, invariant feature-based and 3D model-based methods. As example, a Gabor wavelet-based feature extraction methods was proposed, which is robust to small-angle rotations.

So, we discussed face detection and recognition techniques, problems associated with them and possible solutions.

2.4 Concepts and Object detection

Animals and Plants could be considered as objects. Thus could be detected by statically trained classifiers that normally use interest points or other appropriate features. The VideoFlow project, described in Section 2.1.4, already implements some methods to realize such a recognition.

The basic idea of “interest points” is to find distinctive locations in the image (corners, blobs, etc) and make sure that its detector has a property of repeatability¹⁰. Next, a feature vector represents the neighborhood of the point. This descriptor should be distinctive and robust to noise, geometric and photometric deformations. At the end a distance between feature vectors is calculated, providing in such way a result of similarity. The dimension of the descriptor plays a significant role, bigger descriptor has bigger dimensions which influences computational cost, distinctiveness and robustness.

The Speed-Up Robust Features (SURF), proposed by Hebert Bay et al. [8] uses integral images, which drastically reduce computational time. The integral image is an image representation where each entry at location x (x, y) represents the sum of all pixels in the input image within a rectangular region formed by the origin and x . This representation offers fast computation of the rectangular region intensity: once the integral image has been computed just three additions are needed. The Figure 2.18 shows examples of integral images.

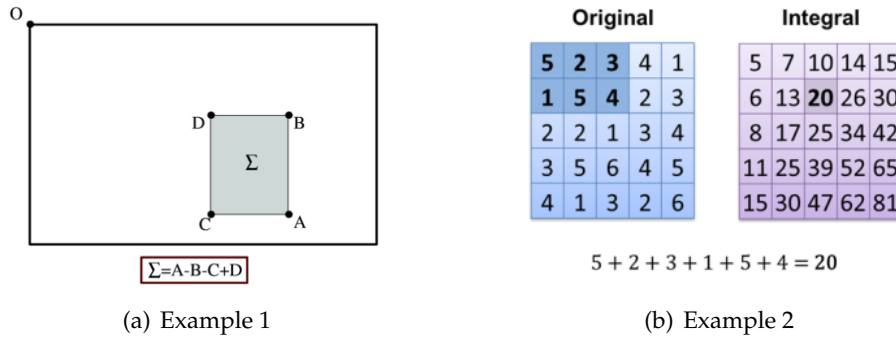


Figure 2.18: Integral images examples

The detector is based on the Hessian matrix, which has good performance and accuracy. Locations with the maximum determinant are selected and detection of the blob-like structures is preceded. Given a point $x = (x, y)$, in an image, the Hessian matrix in x at scale delta is defined as:

$$\mathcal{H}(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xy}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix} \quad (2.2)$$

where $L_{xx}(x, \sigma)$ is the convolution of the Gaussian second order derivative. They proved to be optimal for scale-space analysis. To localize interest points, a non-maximum

¹⁰The property of finding the same physical interest points under different conditions

suppression in a $3 \times 3 \times 3$ neighborhood is applied to the image and over scales. Then the maxima of the determinant of the Hessian matrix is interpolated. The interpolation is specially important because of the difference in scale. Figure 2.19 shows an example of detected interesting points.

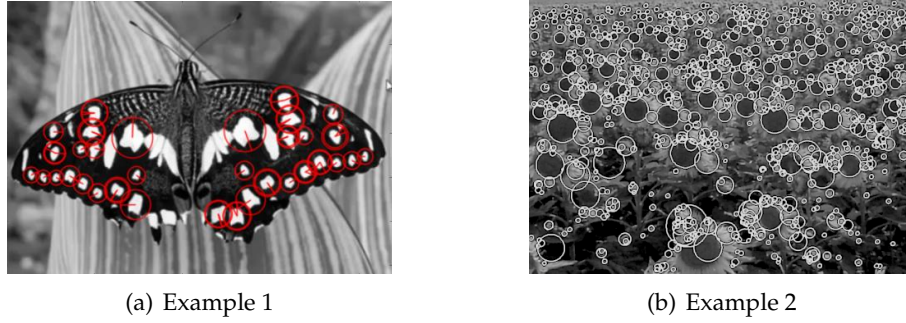


Figure 2.19: SURF keypoints examples

The SURF descriptor use only 64 dimensions. It describes the distribution of the intensity content within the interest point neighbourhood using the first order Haar wavelet responses calculated from integral images. To be invariant to image rotation, a reproducible orientation for the interest points is identified. Then the descriptor extraction process constructs a square region centred around the interest point and oriented along the orientation from the previous step. The region is regularly split into smaller 4×4 square sub-regions where Haar wavelet responses are computed. Computed results are summed up over each subregion, the sum of absolute values too. The concatenation of extracted features for all 4×4 subregions yields a feature vector of length 64. The contrast invariance is achieved with a scale factor by turning the descriptor into a unit vector. Also a contrast invariance is very important for the matching step, feature comparison is proceeded only if interesting points have the same contrast, otherwise the match is discarded. This minimal information allows faster matching, without performance loss of descriptor. The use of smaller regions or greater features is possible, but results are slightly worse or computation cost turns very high. Figure 2.20 illustrates a SURF descriptor. The results slightly outperformed a SIFT method, even with the fact that SURF method does not consider color information.

Peter N. Belhumeur et al. [10] built a hand-held computer vision botanical identification system. The system serves to identify a plant species from the photo of the isolated leaf. The photo needs to be captured with blank background. Thus the system extracts the leaf shape and matches it to the shape of leaves of known species that are stored in the database. In a few seconds, the system displays the top matching species, along with textual descriptions and images.

Another study, made by Neeraj Kumar et al. [43], describes a similar system. It is a mobile application for identifying plant species, called Leafsnap. The recognition process is divided into four steps. First, the image is classified into a valid leaf or not. Next,

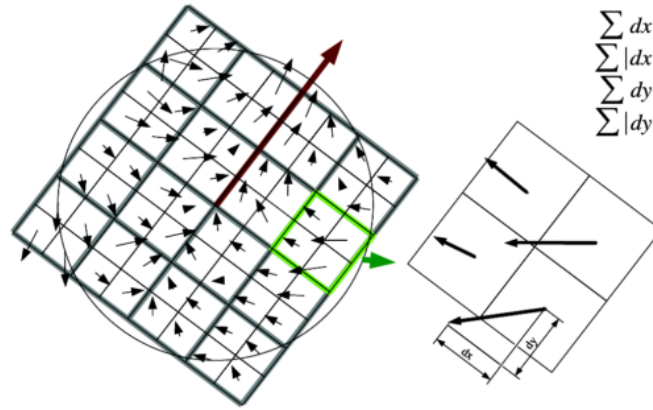


Figure 2.20: SURF descriptor

the segmentation process separates the leaf from the background (background should be solid light-colored). Then the system extracts curvature features from the banalized image for compactly and discriminatively represent the shape of the leaf. Finally the system finds the closes match. The most important improvement of this system in compare to the previous is the use of a simpler and more efficient curvature-based recognition algorithm instead of the Inner Distance Shape Context [55]. The system is available for iPhone and could be installed from the iTunes [17].

Xiaoyuan Yu et al. [116] presented an automated species identification method for wildlife pictures captured by remote camera traps. The analyze of wildlife pictures is more difficult. Generally they have lower frame rates, poor illumination, background clutter, serious occlusion and complex poses of the animals. The proposed method is based on the ScSPM [31]. Firstly, the algorithm densely extracts local feature descriptor. It is a combination of scale-invariant feature transform (SIFT) [57] and cell structured local binary patterns (cLBP) [4]. In order to represent local features, each kind of descriptor feature is learned via weighted sparse coding [102]. Then in order to construct the global image feature vector, max pooling is employed using SPM. To classify images method used a linear SVM. Experiment results showed that a combination of SIFT and cLBP as descriptors of local images features significantly improved the recognition performance.

In 2009 [36] describes an approach for a concept detection. Generally it is based on Regularized Least Squares (RLS) classifiers. Proposed classifiers yield binary classification and are trained with dataset of positive and negative examples. As written in the section 2.1, classifier need a feature set for its training. In proposed approach author used two sets of features: color and texture. The first one is calculated in a Hue Saturation Value (HSV) color space. The image is divided into nine blocks where mean and variance for each channel is extracted. Sometimes exact division may introduce discontinues in objects, as a result a segmentation is done with a Mean-Shift algorithm. It divides image by color regions. The texture features are obtained with Gabor filters.

2.5 Action Identification and Recognition

Human activity recognition is an important area of computer vision research. It is used in surveillance and monitoring systems [32], human-computer interactions and in the filtering of multimedia material like detecting horror [79] or pornographic scenes [35]. In this section we will discuss various state of the art research papers on human activity recognition mostly retrieved through references in the Review of Human Activity Analysis [3].

The goal of this section is to enable a reader to understand the context of the development of human activity recognition and comprehend the advantages and disadvantages of the different approach categories. To achieve this goal J.K. Aggarwal et al. provide a complete overview of the state-of-the-art human activity recognition methodologies. They discuss various types of approaches designed for the recognition of different levels of activities like:

Gestures

Elementary movements of a person's body part: "Stretching an arm", "Raising a leg", "Raising a hand".

Actions

Single-person activities that may be composed of multiple gestures organized temporally: "Walking", "Waving", "Punching".

Interactions

Human activities that involve two or more persons and/or objects: "Two persons fighting", "A person stealing a suitcase from another", "Children playing with the ball".

Group activities

Activities performed by conceptual groups composed of multiple persons and/or objects: "A group of persons marching", "A group having meeting", "Two groups fighting", "Two teams playing football".

Thus, action recognition methodologies are divided into two main groups: Single-layered and Hierarchical approaches. The first one is suitable for the recognition of gestures and actions. The Hierarchical is better for representing high-level activities since it describes them in terms of other simpler activities, sub-events. Each of these two groups has further divisions that are showed at Figure 2.21. We will discuss them in the next sections.

2.5.1 Single-layered Approaches

Single-layered approaches recognize human activities directly from video data. They consider an activity as a particular class of image sequences and recognize the activity

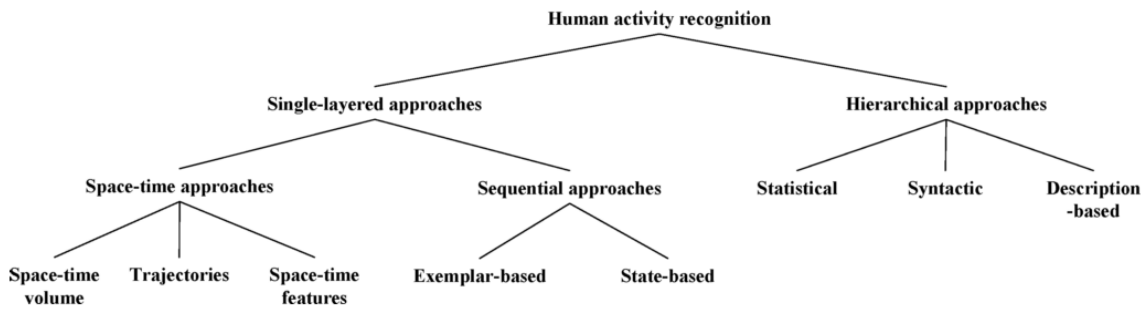


Figure 2.21: Human action recognition approach-based taxonomy

from an unknown image sequence by categorizing it into its class. Algorithms with this approach are most effective when a particular sequential pattern that describes an activity can be captured from training sequences. Most of them use *sliding windows* techniques that classifies all possible subsequences. J.K. Aggarwal and M.S. Ryoo categorize this approach into two classes: space-time and sequential.

The first models a human activity as a particular 3-D volume, in a space-time dimension or a set of features extracted from the volume. This video volumes are constructed by concatenating image frames along a time axis, and are compared in order to measure their similarities. There are several variations of the space-time representation. The system may represent an activity as a trajectory in a space-time dimension or other dimensions. To realize this the system needs to track feature points such as estimated joint positions of a human to draw a trajectory. Another way to describe the space-time volume is the extraction of the set of features. With this approach we are able to describe the activity using local features, local descriptors or interest points. Similar to the object recognition process, the system extracts specific local features that have been designed to capture the local motion information of a person from a 3-D space-time volume. These features are then combined to represent the activities while considering their spatiotemporal relationships or ignoring their relations. Then recognition algorithms are applied to classify the activities, namely template matching, Support Vector Machines or other classifiers.

The second considers an input video as a sequence of observations (feature vectors) and deduce that an activity has occurred in the video if it is possible to observe a particular sequence characterizing the activity. These approaches convert a sequence of images into a sequence of feature vectors by extracting features that describe the status of a person per image frame. The extracted features could be degrees of join angles. Then, the extracted feature vectors are analyzed to measure the behavior of the person that is reflected on the features vector variations. The likelihood between the sequence and activity classes are computed and a decision is produced. Sequences could be extracted directly from the samples or by construction of the trained model. The state model-based approach uses statically trained models that correspond to sequences of feature vectors

belonging to specific activity class. Normally one model corresponds to one activity and has a set of states that characterizes this activity.

2.5.2 Hierarchical Approaches

It is very difficult to recognize complex human actions using Single-layered approaches, hence, Hierarchical approaches appeared. They recognize high-level activities (for ex: "Fighting") by merging low-level ones (in the case of "Fighting" merging "Punching" and "Kicking"). In other words, the recognition process of high-level activities consists in hierarchical concatenation of atomic-level actions (common activity patterns of motion, frequently appeared during high-level human activities). The atomic-level action normally are recognized using single-layered approaches described above.

In the domain of the high-level actions recognition approaches hierarchical methods are the most suitable. Their main advantage over non-hierarchical approaches is the ability to work with more complex structures and semantic-level analysis of interactions. They could be trained with less data and incorporate prior knowledge into the representation (listings of atomic-level actions and relations between them).

J.K. Aggarwal and M.S. Ryoo categorize hierarchical approaches into three groups: statistical, syntactic and description-based.

Statistical approaches

This approach uses multiple layers of state-based models (mostly Hidden Markov Models [111]). Considering two-layer structures, at the bottom layer, atomic action are recognized from feature vectors and converted into atomic action sequences. The second layer models treat these sequences as observations generated by the second-level models. Then for each model a probability of the model generating a sequence of atomic-level action is calculated. The obtained probability value is used to measure the likelihood between the stored activity and the input image sequence.

Syntactic approaches

Syntactic approaches based on the context-free grammar programming. They model human activity as a string of symbols. Each symbol corresponds to an atomic-level action. Hence, a set of production rules generates a string of atomic actions which is recognized by adopting parsing techniques from the field of logic programming languages. One of the drawbacks of this approach is the recognition of the activities composed by concurrent events such as "sub-event1 must occur during sub-event2". In this type of approaches atomic-level activities must be strictly sequential.

Description-based approaches

Unlike previous approaches, description-based are able to handle activities with concurrent structures. This ability is achieved by the manner of human activity

representation used in these approaches. This type of approaches treats human activity in terms of simpler activities describing their temporal, spatial and logical relations. They widely use Allen's temporal predicates (*before*, *meets*, *during*, *finishes* [5]) to specify relationships between time intervals. Generally, the recognition is performed by developing an approximation algorithm that solves the constraints satisfaction problem.

2.5.3 Existing Methods

There are many methods developed. Figure 2.22 compares the classification accuracies of some of them on the KTH dataset. This test dataset [83] is designed to test general-purpose action recognition systems academically. It contains videos of different participants performing simple actions such as walking and waving, which are taken by the authors in a controlled environment [3].

We could see that the best ones were: [80] and [52]. Both are categorized as a Single-layered, space-time local feature approaches 2.5.1.

Ryoo and Aggarwal [80] introduced the spatiotemporal relationship match (STR match), which is designed to measure structural similarity between sets of features extracted from two videos, enabling the detection and localization of complex-structured activities. Their system not only classified simple actions (i.e., those from the KTH datasets), but also recognized interaction-level activities (e.g., hand-shaking and pushing) from continuous videos.

Z. Li et al. [52] present a luminance field manifold trajectory analysis based solution for human activity recognition, without explicit object level information extraction and understanding. The proposed method is computationally efficient and can operate in real time.

To close this section we can conclude that described related work above shows a good review of the existing methods, techniques and state-of-the-art methods. With realized study we can proceed into developing process. Next section describes it.

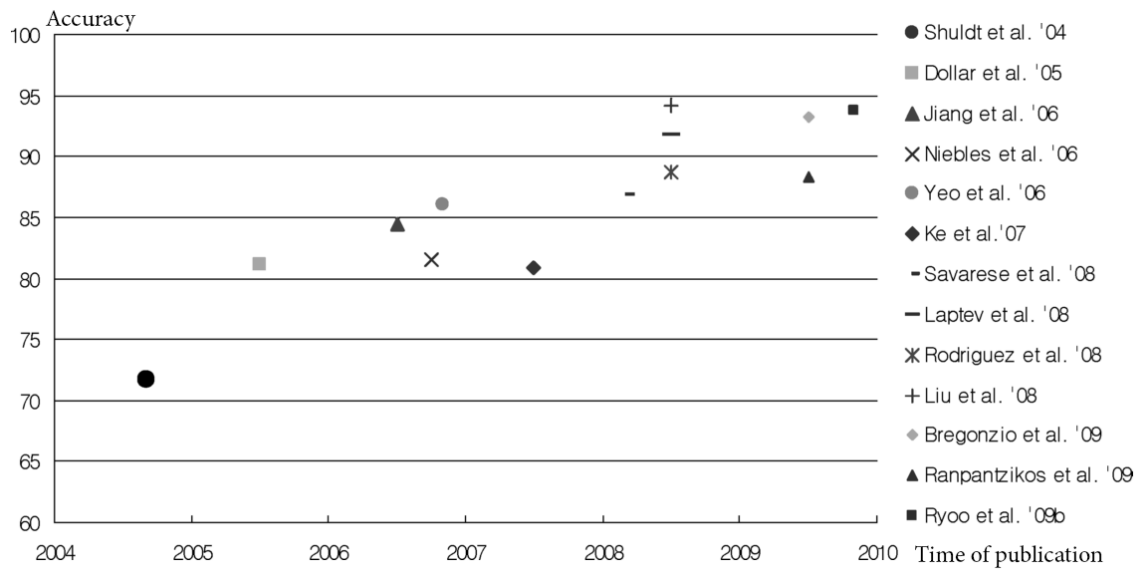


Figure 2.22: The classification accuracies of human action detection systems tested on the KTH dataset.



Solution

In this chapter we will discuss the realized solution. The solution is based on previously discussed related work and will be integrated in a video-mail system called VeedMee.

The goal of the project VeedMind is to develop multimedia information search capabilities in the video-mail system. Since any mail system suffers from overload (the problem of a large volume of incoming mails) the system presented below offers techniques that extract multimedia information which will be used for multimedia search and content management:

- Video segmentation
- People detection and recognition
- Scene text detection and recognition
- Concepts detection
- Image-based search
- Action detection and recognition

For a better understanding of these techniques and their contribute to the system, subsection 3.1 includes descriptions and diagrams of the VeedMee system and VeedMind integration. Section 3.2 explains an implementation of the VeedMind components, as well as its sub-components that implement techniques mentioned above. Finally, section 3.3 shows obtained testing results.

3.1 Design

This section explains the design of the system. Subsection 3.1.1 shows the use case model of the VeedMind system, then VeedMee architecture is presented with the integration of VeedMind component in Subsection 3.1.2 and finally Subsection 3.1.3 shows the VeedMind architecture.

3.1.1 Use cases

The use case diagram describes how the system will be used, essentially on the user perspective. Since the realized system is a component of a larger one (VeedMee), the user will not be a real person, it will be other system components that call required services.

VeedMee is a video-mail system, where the main purpose is to use multimedia material, specially video, as a way to communicate. Besides this, the system should offer additional services to make a good user experience. As a result the main services required from the VeedMind system are related to:

- Preview of the video (thumbnails)
- In video search
- Advertisement suggestion
- Content alignment

The diagram on the Figure 3.1 shows the use cases. There are two actors: Services (component from VeedMee) and Admin (person who will manage and configure VeedMind). The description of each use case comes below:

Submit video

The Services submits a video file for analysis. The video file is transferred into the VeedMind hosting machine. Then the system begins analysis of the file, extracts metadata and intermediate results are stored in the system. The system responds with notification of successful submitting of the file.

Submit faces DB

In order to realize face recognition, the system needs a trained database of known faces. There are two possibilities of submitting the database: upload images for training or using an already trained database file. The system trains or loads the training database. As a result, it is possible to run the face recognition process.

Get thumbnails

The VeedMee system aims to offer to the user fast preview of the video. More, to improve user experience, VeedMee offers a fast orientation between frames. To

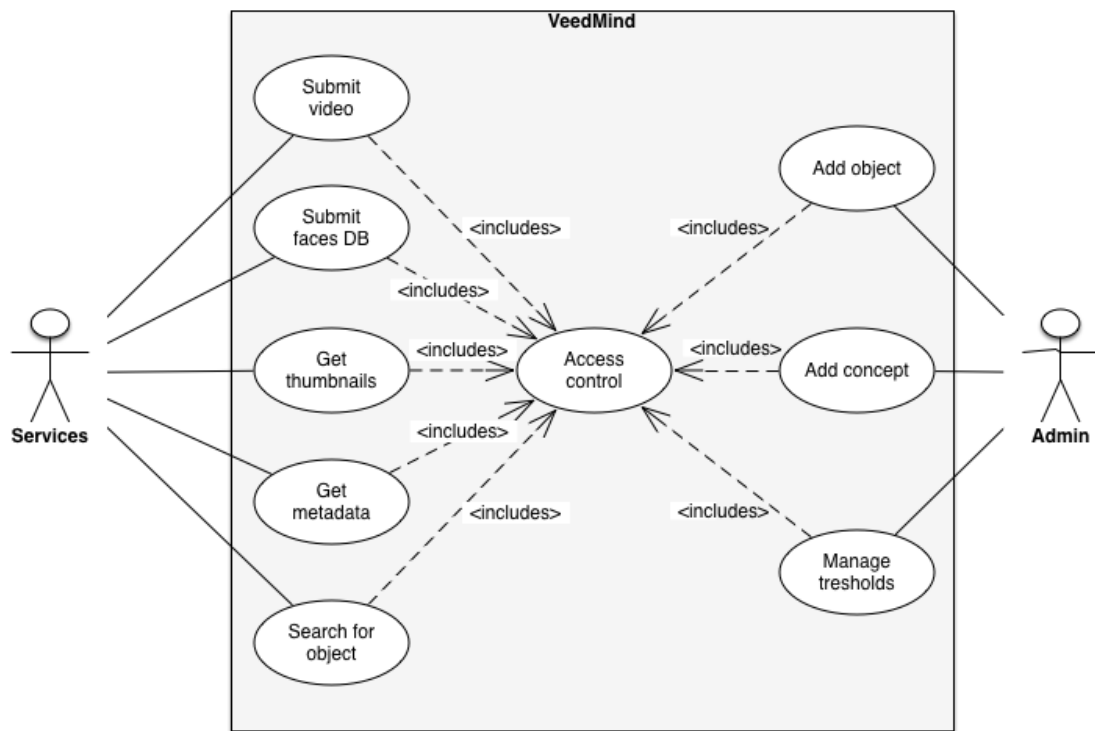


Figure 3.1: VeedMind use case diagram

realize it VeedMind calculates variations between frames and selects the most suitable, usually called key-frames. The process calls Segmentation and will be better described in the next section. Segmentation produces a XML file that contains indexes of the key-frames.

Get metadata

At the end of video file analysis, VeedMind yields a final XML file which contains all metadata retrieved from the video. This metadata is composed by detected and/or recognized faces, objects, concepts and actions with the instant where they appear. The structure of the XML file will be described in Section 3.2. The final XML file is returned as an answer to the incoming get request.

Search for object

If a determined user wants to search for a personal object (unknown by the system) it submits a new query with an example. The query is redirected to VeedMind. The system extracts features from the example and compares to the extracted ones from processed video frames. If something is found it is returned to the Services and user as well.

Add object

The system should be growing with time, thus a possibility of adding new objects

is offered. The admin of the system can submit examples of the new one. VeedMind extracts features from examples, stores them and adds into a local database of known objects. The object itself is an image that could appear in background of the scene. A good example of it is a mobile phone or brand logo.

Add concept

Equally to the objects, the possibility of adding new concepts should be supported. Concepts are characterized by two sets of features and trained classifier. To train the classifier the training set should be provided with positive and negative example images (images with and without concept). The system receives the training set, trains a classifier and stores it into the local database. In practice the concept is a whole scene, example: indoor, office, presentation or car.

Manage thresholds

Each subcomponent of the system could be parametrized. This use case is exactly defined for this. An admin could redefine thresholds that influence segmentation parameters, face and text recognition accuracy, minimum and maximum sizes as well as an object detection precision.

Access control

Every access into the system should be controlled. This use case provides such control. As a result only authenticated users can interact with the system.

3.1.2 VeedMee architecture

Figure 3.2 shows the proposed system integration architecture. The VeedMee system is hosted on a big datacenter. VeedMind is a stand-alone component. It will work as a Web Service which receives calls from other components and produces results, that will be stored in the database.

VeedMee works basing on the Model View Controller (MVC) architecture. The user accesses the system from the internet (default is a Web browser). As the figure above shows, VeedMee has six major components:

UI

User interface, considering MVC architecture, this component is equivalent to the View. Thus it deals with user requests and sends them to controller. In practice, the UI component is a web page (www.veedmee.com), where can compose messages, record videos, visualize them and of course send and receive composed multimedia messages to other users. This is the core functionalities of VeedMee, apart from them the user can visualize advertisement and manage his archive of messages and contacts.

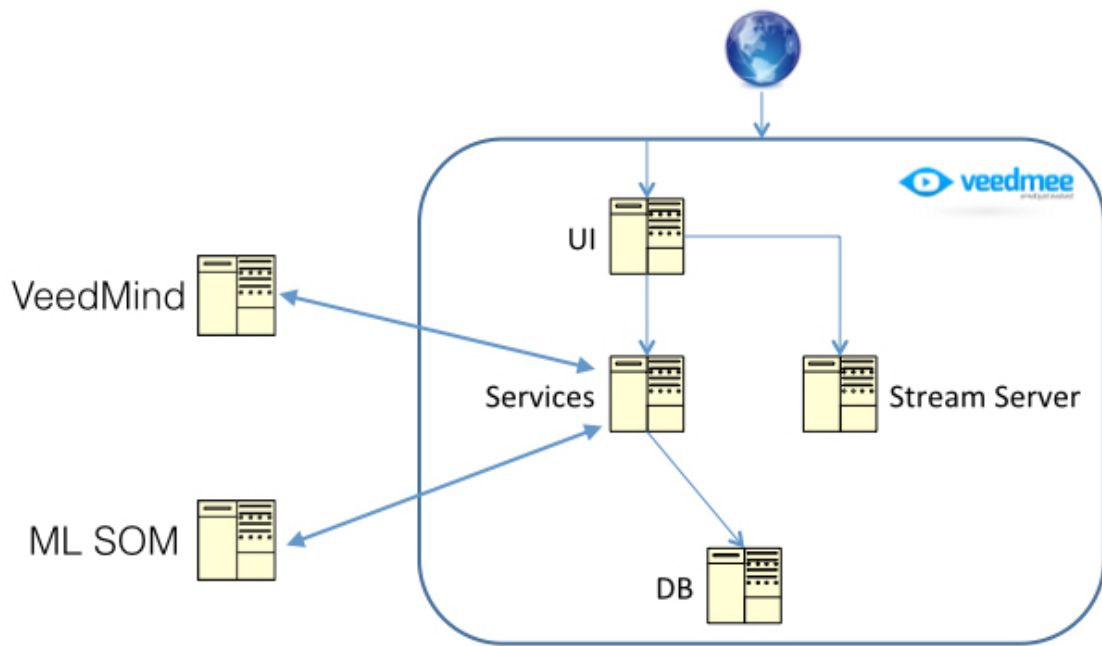


Figure 3.2: VeedMind integration architecture

Services

This component manages a whole system. Hence in MVC classification it is a controller. It deals with requests received from a view component, processes them, responds and stores results into a database. As an example the user wants to send a video mail message: from a web browser the user creates this request. He records a video, which is stored in a Stream Server, writes recipient's address, text, etc. The created request is sent to Services, which stores related metadata into a database and processes a submitted request in order to notify recipients. As a result the UI shows to user successful confirmation of request processing and the recipient receives the notification of a new message.

Stream Server

As we mentioned before, this component stores multimedia material recorded or uploaded by users. Furthermore, it streams videos into UI component which shows them to user. Since it is designed for storage it could be called a model.

DB

Database, this is a well-known model component. It stores the whole data of the system and its users. Since VeedMee is a commercial product with real users, we cannot show and describe a database structure and tables.

VeedMind

This is our component. It is designed for a video processing, producing metadata

which is stored into the DB. In order to separate computation and for security reasons, this component along with ML SOM stands out from VeedMee system and acts as a Web Service. We will describe it in more detail in the next section.

ML SOM

This is the machine learning component. It processes results that come from VeedMind to find semantic meaning. It has an access to the user's information and tries to find the best semantical meaning of extracted multimedia metadata focusing on the user.

3.1.3 VeedMind architecture

The VeedMind system prototype is organized as stand-alone component. It communicates with exterior via Web Services. The components diagram below shows how the system is constructed. The explanation of the each one follows below:

Video Processing System

The user interface, considering the MVC architecture. This component is equivalent to the View. Thus it deals with user requests and sends them to controller. In practice, the UI component is a web page (www.veedmee.com), where the user can compose messages, record videos, visualize them and of course send and receive composed multimedia messages to other users. This are the core functionalities of VeedMee, apart from them the user can visualize advertisement and manage his archive of messages and contacts.

Segmentation

This component receives a video clip and returns the list of relevant cut frames. These frames are shots/cuts of the clip where there is a large difference in the scene and they are used to describe the whole clip.

The performance of this component is proportional to the size of the video clip. Currently, it scans every pixel of every frame. Thus the videos with a high resolution and long duration will take more time to be processed. The number of detected cut frames depends of video content. If it is very uniform (same environment, same objects...) a small number of frames will be produced.

The component has been developed. The segmentation is done in three main steps:

1. Retrieving frames differences — for each frame it calculates the histograms and compares them to the ones of the next frame. The produced values of the differences between sets is stored as a descriptor of “how much different are the frames”.

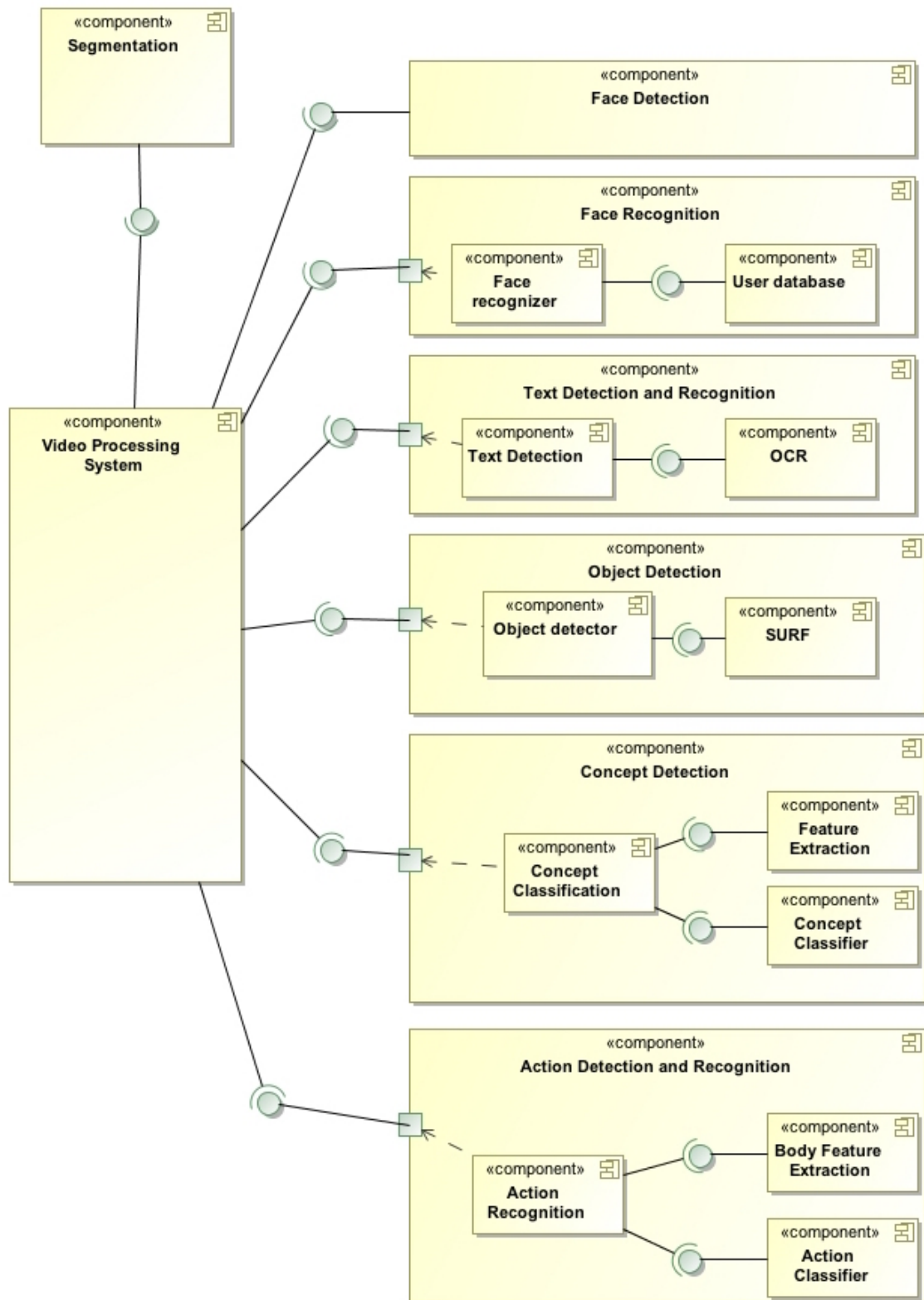


Figure 3.3: VeedMind component diagramm

2. Filtering — frames with a small difference will be discarded. Beside this, each 25th frame is quickly scanned for the possible presence of faces. If there are faces the frame is not discarded, even with a small difference from the previous frame.
3. Extraction — filtered frames are stored for future processing.

The segmentation process reduces processing weight, since it reduces the volume of data to analyze. For example, from 819 total frames we analyze only 34 or from 3290 we analyze 77. Thus, in these examples we reduced the input data to 4.15% and 2.34% respectively.

Face Detection

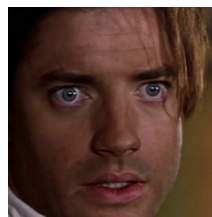
This component receives a video and a list of selected frames. At the end it returns the list of detected faces. Each element of the list specifies the frame index within a video. The list is iterated, the video is set into selected frame and images could be retrieved. Retrieved image are processed with a face detection algorithm. The algorithm detects only front positioned faces, since we need only this type for further recognition, figure 3.4(a) and 3.4(b) shows an example. It uses Haar Cascades classifiers, proposed by Paul Viola and Michael Jones in 2001. The frames with faces are marked with a presence of this concept. The related metadata is stored in the system.

Face Recognition

It recognizes faces previously detected by Face Detection. The metadata stored in the system indicates which frames contain faces. These frames are retrieved from the video. The face is cut from the image and normalized as a stored training set. The normalization process uses the eyes as a reference to scale, rotate and cut the face. Then the cut is processed with special filters that remove excessive sharpness and uneven illumination, figure 3.4(c). At the end, a face is predicted with an Eigenfaces algorithm. If the resulting values of unknown person and confidence are below thresholds, the face is marked as recognized and the respective id is associated to metadata. Otherwise, it is marked as unknown.



(a) Scene with face



(b) Cut face



(c) Normalized face

Figure 3.4: Face detection and normalization

Text Detection and Recognition

As the previous module, it receives a frame and returns the list of identified text elements on it. The component is divided into two main parts: Text Detection and OCR. The first one identifies the regions with text within the scene, crops and normalizes them. Next they are handled by the Optical Character Recognition, and for this the Tesseract OCR is being used. Currently it recognizes by default an English language text, but it supports other languages as well. At the end, recognized text is added into final XML file.

The component has been developed and works. Text detection is done with an OpenCV 3.0 Class-specific Extremal Regions for Scene Text Detection. It detects presence of text and gives a bounding box coordinates. Then detected box is cut and normalized, figure 3.5. Normalized image is passed into Tesseract OCR and recognized.

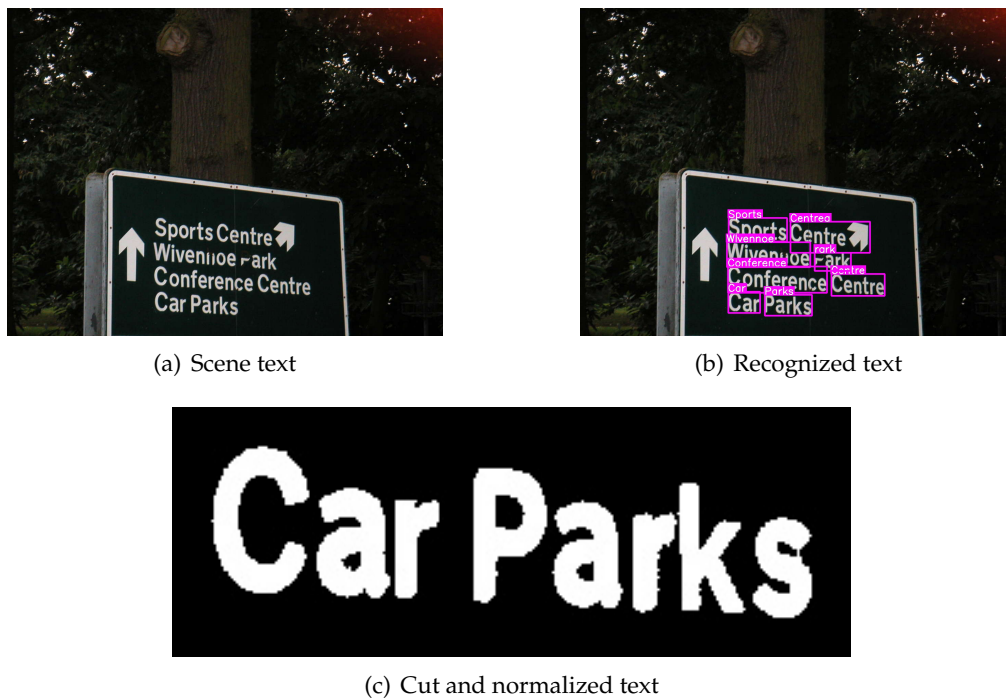


Figure 3.5: Text detection and normalization

Object Detection

It analyses a frame and extracts the points that are independent of scale and rotation conditions and textures. The SURF (Speeded-Up Robust Features) algorithm is used for this phase. Extracted key-points and descriptors are matched with the template ones, present in the system. Matches are filtered, thus only a sufficient amount of equal key-points with a small distance between them will be passed. This means we have an object present on the scene. Then, a possible homography is computed to find a location.

Concept Detection

This component analyses a frame and extracts the color moments and texture features. Extracted information is used to ask a classifier if a scene corresponds to the classifier's concept or not. Each classifier is previously trained with a training set, which contains examples of each concept.

Action Detection and Recognition

This component is being developed. The basic idea is to identify human body parts and track them between sequential frames.

3.2 Implementation

This section will describe functionalities and the workflow of the VeedMind system. In order to make possible fast and simple future updates, it was constructed with separated independent modules (components). Each module could be hosted in different machine and be executed in parallel for different videos. Some components could be parallelized even internally. The flow chart diagram 3.6 shows the workflow of the system and intermediate files. Each component will be described below.

Video Processing System

The functionalities of this component were described in the previous section. In terms of implementation it is a C++ Class which reads a system files folder to find video files in it. Then each video file is sequentially loaded and processed by the system. Loading a file means reading it from the system and create internal structure for its processing. Inside the system video the file is transformed into a "VPFSVideo". VPFSVideo contains a video file object, segmentation information, standard video details (duration, frames, fps...) and other variables like processing time. The segmentation information is characterized by threshold values and lists of shots (complete and filtered). To get best results the video could be segmented more than one time and as a result threshold information should be stored.

The Video Processing System is the core of the whole system. After creating a VPFSVideo and done its segmentation, VPFSVideo object is passed into other modules for future processing. It stores intermediate results and produces the XML files that contain final results of the processing. The XML file Cuts is an intermediate file produced by the Segmentation in order to provide thumbnails for the User Interface of the system.

Segmentation

The role of this component is already known. An algorithm of the absolute difference of histograms is used. To simplify computation cost each histogram is constructed from the black and white image. The difference is calculated from the sum of all pixels of the neighbor frames. If the calculated result is above threshold

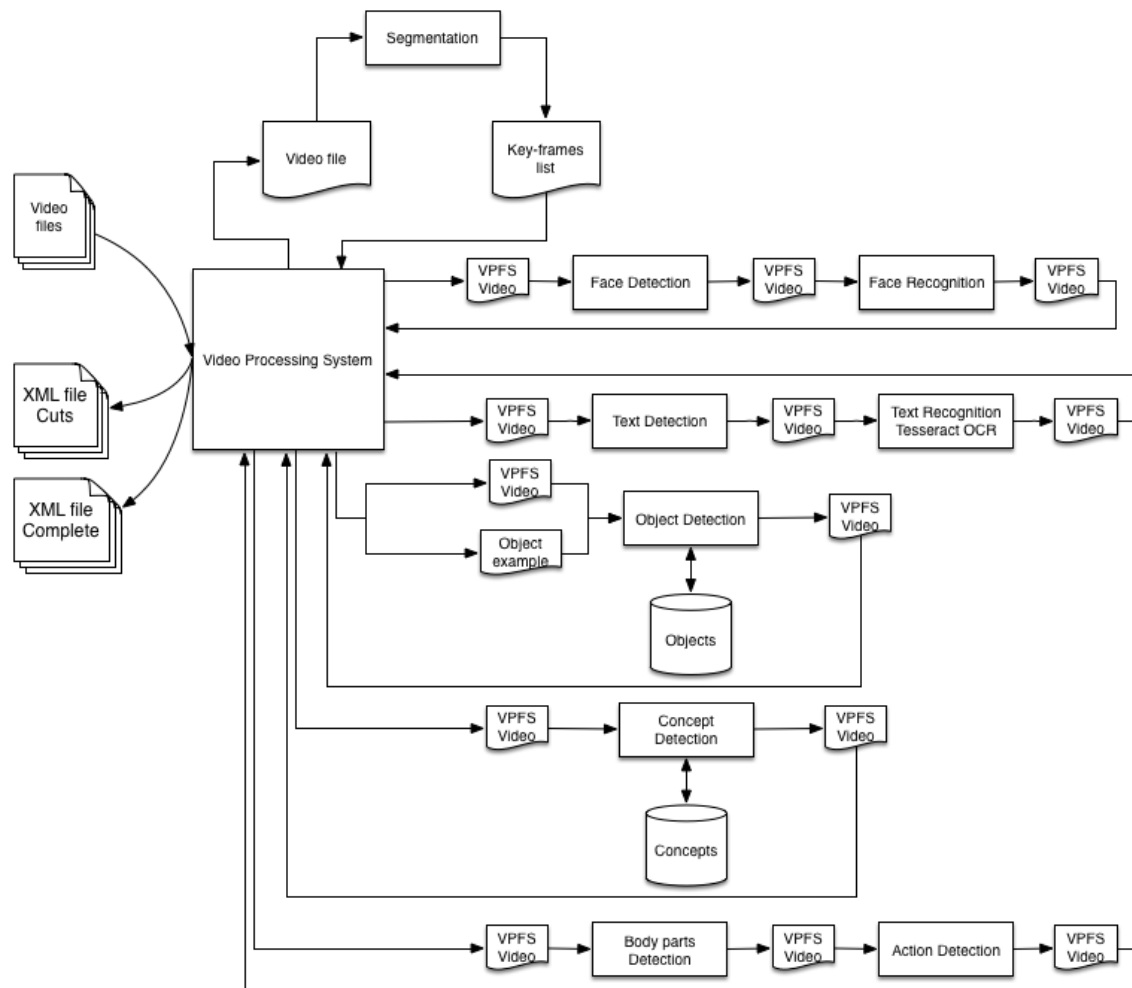


Figure 3.6: VeedMind workflow diagram

the shot cut is obtained and the frame between two sequential cuts is marked as a key-frame. The threshold could be readjusted depending on the video capture conditions (ex: fixed camera or not). The result of segmentation is stored as list of shots (VPFSShot) which contains a key-frame index and other related information. To avoid information loss, the segmentation module has an additional filtering, each 25th frame is quickly scanned for a presence of human face. It is important for future face recognition due to the face position and illumination variation.

Face Detection

Face detection is done with the algorithm proposed by Paul Viola and Michael Jones in 2001 [101]. It is explained in the section 2.3. The face detector could be configured to discard too small or too big faces. Detected faces metadata is stored into VPFSShot as a "Face" concept.

Face Recognition

After faces have been detected, the list of VPFSShots contains related information. This list is iterated and if VPFSShot contains a "Face" concept, it is cut and passed into the recognizer. As it was described below, the recognizer normalizes a face into appropriate format of the system. The first step of normalization finds eyes coordinates. There are two algorithms employed for this task, the first one uses Haar Cascade classifier with eyes cascades training values. If the cascade classifier does not find two eyes, the means of gradients algorithm is run [95]. Having eyes coordinates, the distance between them is measured and the face is scaled and rotated. Then in order to remove excessive sharpness and illumination effects, the face is filtered with bilateral filter and histogram equalization. Recognition itself is done with the Eigenfaces algorithm. It falls into the PCA approaches category explained in section 2.3.

Text Detection

Text detection is done with the recent Class-specific Extremal Region algorithm proposed by Lukás Neumann & Jiri Matas. To increase text sharpness, a frame is processed with a simple Gaussian filter. Then it is passed into the algorithm. The algorithm uses a sequential classifier trained for character detection. The classifier processes, step-by-step, a thresholded component tree of an image. To improve the response time and efficiency, the algorithm is divided into two phases. The first incrementally computes descriptors and estimates the class-conditional probability. Filtered output of the first stage is passed into a second. It classifies regions into character and non-characters classes using more complex and more computationally expensive features.

Finally, character classes are grouped. It is done with the Lluís Gomez and Dimosthenis Karatzas algorithm. The algorithm finds meaningful groups of regions using

perceptual organization and combines two different clustering techniques. Both algorithms are described at the end of the section 2.2.1.1.

Detected text is cut and automatically passed into the OCR.

Text Recognition

Optical Character Recognition is offered in OpenCV 3.0 API. It can be done with two already implemented methods: OCR Tesseract and OCR HMM Decoder. The first one is described in section 2.2.2, the second one uses Hidden Markov Models to realize OCR. Because of the performance and configuration possibilities we use Tesseract. If OCR recognizes the text on the cut, text information is stored into VPFSShot previously passed into Text detector.

Object Detection

The base of this component is the use of SURF descriptors and their matching explained in section 2.4. The database of objects stores their key-points and descriptors. To find a presence of the object, each object's descriptor is compared to extracted descriptors from the frame. If the match is found, key-points comparison is done in order to find object's coordinates. If the user wants to find its own object, he gives an example image. Equally to the frame features extraction, SURF features are extracted from the user's example and compared to descriptors from all frames previously analyzed. Similarly to the previous components, if the presence of object occurs, it is stored as "Object" concept into the appropriate VPFSShot.

Concept Detection

Concepts are detected based on the color and texture features explained in section 2.4. The database of concepts stores previously trained SVM classifiers. Each classifier was trained with a specific set of features (color, texture or both) to test the concept presence on the input frame. The same set of features is extracted from the input and then passed to the concept's classifier. The classifier tests input features and yields a positive or negative result. If the result is positive, the concept is pushed into VPFSShot's list of detected concepts as a "Concept" concept.

Action

Since we are developing this component, the basic idea is to extract features (human body parts), track them in order to create a bigger feature vector which will be a template for each action. Then the problem is similar to the concept detection. Previously trained action classifiers will be stored in the system and the input feature vectors will be tested in a strategy one against all.

The system provides an output of two XML files. The first one stores segmentation results. Basically the file contains segmentation parameters (thresholds) and a list of shots (key-frames). Figure 3.7 shows the structures of the file.

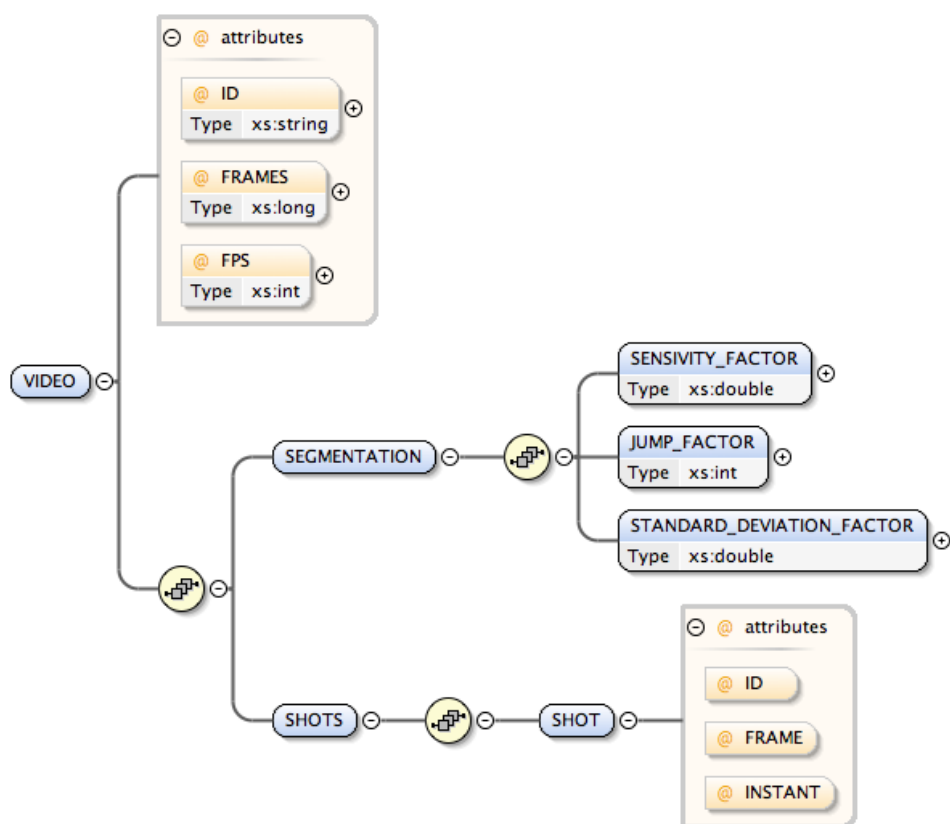


Figure 3.7: Segmentation XML file

The previous file makes a part of the second. This final XML file contains information about each key-frame, namely position (frames count), instant and detected concepts. Each concept is characterized by its type and other information depending on its nature. For text it is a position and content, equally for the object, where content is a name of it. For face the content is characterized by the name and email of recognized person. The concept of type concept contains just the name of the concept. Figures below shows the structure and example of the final file.

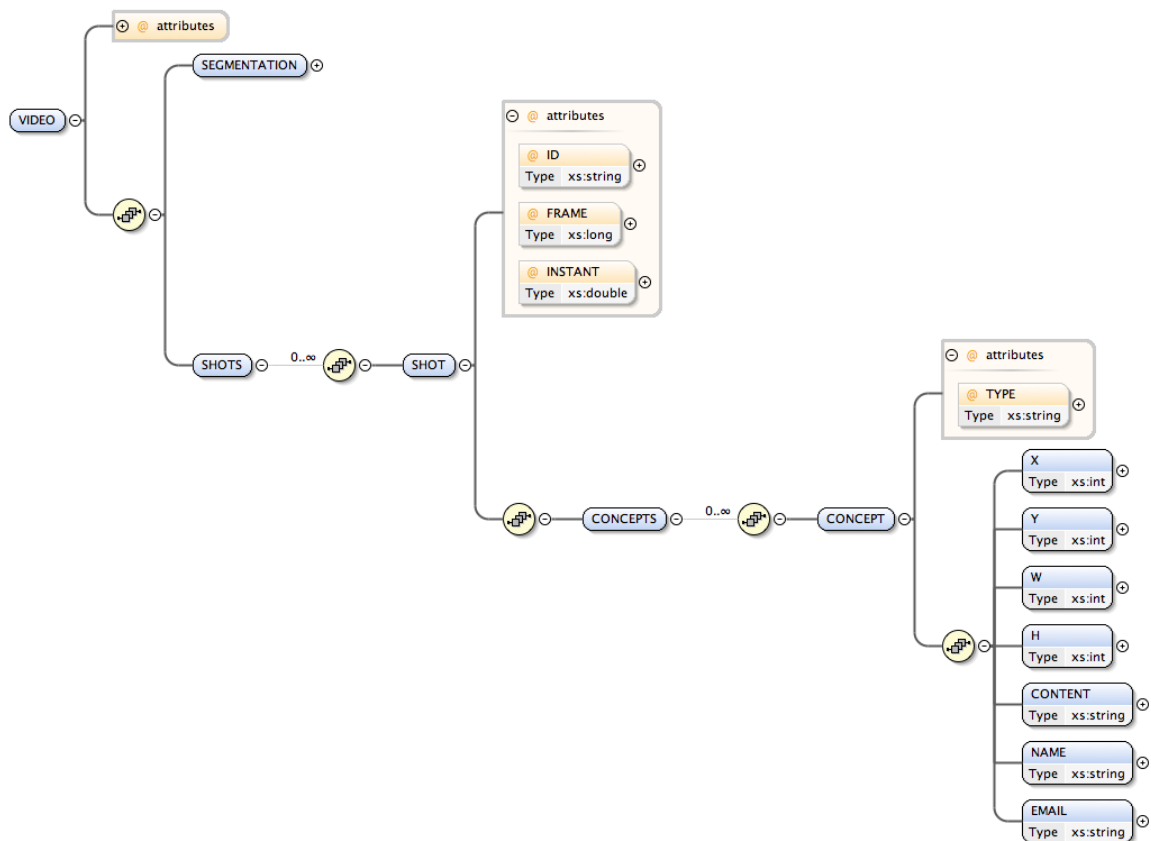


Figure 3.8: Final XML file

Listing 3.1: Produced Example XML

```

1 <VIDEO ID="video0001" FRAMES="15339" FPS="25">
2   <SEGMENTATION>
3     <SENSIVITY_FACTOR>1.700000000</SENSIVITY_FACTOR>
4     <JUMP_FACTOR>10</JUMP_FACTOR>
5     <STANDARD_DEVIATION_FACTOR>1.000000000</STANDARD_DEVIATION_FACTOR>
6   </SEGMENTATION>
7   <SHOTS>
8     <SHOT ID="0" FRAMES="60" INSTANT="2.4">
9       <CONCEPTS>
10        <CONCEPT TYPE="TEXT">
11          <X>101</X>
12          <Y>295</Y>
13          <W>42</W>
14          <H>13</H>
15          <CONTENT>text content</CONTENT>
16        </CONCEPT>
17        <CONCEPT TYPE="FACE">
18          <X>200</X>
19          <Y>427</Y>
20          <W>150</W>
21          <H>150</H>
22          <NAME>NAME OF PERSON</NAME>
23          <EMAIL>EMAIL@OF.PERSON</EMAIL>
24        </CONCEPT>
25        <CONCEPT TYPE="OBJECT">
26          <X>85</X>
27          <Y>307</Y>
28          <W>86</W>
29          <H>19</H>
30          <CONTENT>NAME OF OBJECT</CONTENT>
31        </CONCEPT>
32        <CONCEPT TYPE="CONCEPT">
33          <CONTENT>NAME OF CONCEPT</CONTENT>
34        </CONCEPT>
35      </CONCEPTS>
36    </SHOT>
37    <SHOT ID="1" FRAMES="126" INSTANT="5.04">
38      ...
39    </SHOT>
40    ...
41  </SHOTS>
42</VIDEO>

```

3.3 Results

This section contains results obtained from prototype testing. The prototype was developed on Ubuntu 14.04 and Eclipse IDE. Both performance and results are presented below.

3.3.1 Discussion

Each feature was tested and results are analyzed and discussed next.

Segmentation

The essential function of segmentation is to reduce a number of future computations. This is the start function of the whole system, as a result it is important to guarantee that produced key-frames list is not too big or too small. The tests showed a significant reduction and good quality of "cut detection"¹. Table 3.1 shows examples.

Table 3.1: Segmentation results of testing videos

Resolution (px)	Duration (s)	Frames total (#)	Frames selected (#)	Reduction (%)
1280 x 720	198.92	4973	69	98.62
640 x 360	11.36	341	12	96.48
480 x 360	593.36	14834	201	98.65

Face detection

The algorithm produces excellent results with low false positive rate. Misclassifications that appear are discarded by the face recognition module, which detects dissimilarity between input and a face object. The face detector could be parametrized with the size of the detection object, thus we realized some tests with different parameters. The database for testing has 916 images with different localizations and conditions. The results were above 90 % what shows a perfect performance.

Face recognition

Face recognition is difficult to evaluate due to the fact of user's unpredictability. As we discussed before the face could appear in different condition which drastically influence recognizer performance. We realized tests with our own videos and a user database. The results showed a good accuracy, but we checked that training is very important in this module. Using more faces offers better accuracy.

Text detection and recognition

As we said before, we use text detection algorithm proposed by L. Neumann et al. [69] in 2012, section 2.2.1.1. Testing results on text detection dataset ICDAR proved a good accuracy of the approach: recall of 64.7 %, precision of 73.1% and the

¹Detection of significant scene variation

f-measure of 68.7%. During testing results on our own video dataset, we checked that the algorithm successfully identifies text regions. An OCR, Tesseract shows a good behavior too. The text regions are recognized, sometimes with some misclassifications because of blurry images, but since we work with video, we can analyze neighbor frames and merge results.

Object detection

The use of SURF descriptors and matching showed excellent results in search of objects. Better precision could be parametrized by the user. Results were obtained with logos of the companies.

Concepts detection

We tested 5 concepts. Each concept was trained and tested with 1404 images. As you can see on the table 3.2, good results are achieved.

Table 3.2: Concept detection results

Concept name	True positives (#)	True negatives (#)	False positives (#)	False negatives (#)
Airplane	8	1387	9	0
Car	250	1149	3	2
Flower	727	655	9	13
Leaf	68	1328	2	6
Motorbike	314	1057	17	16

3.3.2 Performance

Performance tests were done on a MacBook Pro (13-inch, Early 2011) portable computer with the following characteristics:

- OS: Ubuntu 14.04 64-Bit
- Processor: 2,3 GHz Intel Core i5, (2 Cores)
- RAM: 8 GB 1333 MHz DDR3
- HDD: 320 GB, 5400 rpm

To realize tests we used our own video dataset. The dataset contained 8 videos with different durations and resolutions. The table 3.3 presents obtained times of each task, tasks that do not have values means that its value depends to the input samples.

The segmentation explores the frame redundancy property, it reduces frames to process as well as the whole computational cost of the system. Computation of the histogram difference between two images directly depends on their resolution. Bigger resolution brings more pixels to analyze.

Text detection depends on the image resolution and image complexity. Images that contain many edges will take more processing time because of classifier assumption of

Table 3.3: Task testing times

Task name	Time
Frame histogram calculation	9.7 ms
Face detection	3.57 ms
Face recognition training (per training image)	-
Face recognition	-
Text recognition:	30.78 ms
Descriptor extraction of the object	0.36 s
Descriptor matching	-
Feature extraction	0.91 s
Classifier training	18.923 s
Concept prediction	83 ms

possible text candidates. Then each candidate is checked with more precision, which has a higher cost. After detection, cut regions are passed into the OCR. More regions imply more export from the OCR. As a result, recognition time depends of the number of text regions.

The search for an object takes a significant amount of time. The matching processor is quite expensive because of the number of detected key-points. Another factor that influences drastically a search is the dimension of the searchable library.

Concepts detection is simpler. After features extraction, features are tested with each concept's classifiers. More concepts mean using more classifiers.

4

Conclusions and Future Work

This chapter describes conclusions taken from the realized work in this master thesis and some topics for future work. The system was built in a modular way and that opens the door for future improvements.

4.1 Conclusions

In this work we have developed a system which analyzes videos and produces meta-data for each one. The prototype is the first version of the system. The prototype was developed in C++ with the OpenCV and openFrameworks libraries. These libraries are supported on other major operating systems which makes possible a fast and simple integration. This prototype is being integrated into a real production product called VeedMee.

In the related work we have seen many studies about multimedia metadata extraction systems and techniques for computer vision. A detailed review of the text detection was made. This is quite a recent area in computer vision because of its complexity. We proceeded into selection of the best ones in terms of quality/cost. Many techniques were selected because of their existing implementation in frameworks that we use. All selected techniques could be parametrized to suit the context of the problem. We selected the best parameters to obtain the best possible results.

The discussion of functionalities gave us a good view of the system results. It is clear that segmentation reduces the redundancy of the frames but there are some problems like blurred key-frames. With these problems it reduces computation cost and produces a sufficiently acceptable results. Face detection works with high accuracy, so this task is considered as successfully done. Face recognition continues to be a difficult task. Recent

approaches use 3D face models, but such type of approach requires too much training data and computational power. Our approach offers good face recognition results, as we made robust filters in order to prevent wrong classification. The text detection works well, and some false negatives are filtered with the OCR. If the OCR does not recognize anything, there is a big probability of the region not containing a text or it having poor quality. Object detection uses key-point matching and it works but it could be improved with more scene segmentation techniques. Concept detection performs well as shown by the results.

The whole system showed a good performance and could represent a good value as a product.

4.2 Future Work

Any technological product has a limited lifetime. Each year the hardware is updated and new programming software approaches appear.

For our work, the first future step is to integrate an action detector module, which is being developed now. The segmentation module could be improved with better features like edges differences and blur detection. With more powerful hardware it is possible to implement a 3D face detector which will detect faces in different positions and will construct a 3D model. Then this 3D model can be used in the recognition module which can match exactly the correct part of the face or even a 3D model constructed from a sequence of frames. Image quality plays an important role in text detection. Hence, a good improvement is to generate a sharper image or select it from a sequence of frames. The OCR has a good accuracy if the input image has sharp characters. Object detection is quite limited because of variability of key-points. To reduce computations and improve accuracy it is possible to add some kind of edge and shape detector. After detection of a shape, matching a limited region is much faster than the matching of the whole image key-points features. As a result a N-phase cascade object detector could be used. Concept detection showed good results for some concepts, but other concepts need another type of features. Providing different type of features for different type of concepts could improve results. Another good improvement is to use localization of the video to reduce the number of searched concepts. For example if we recorded our video at home, there is no need of searching for airplanes.

In terms of computation, the system could be parallelized among different machines. Because of the computational cost differences of each module it is better to separate processing in order to over pass CPU bound operations and get the best performance. Another possible improvement of performance would be to use different resolution videos for different modules.

Bibliography

- [1] 9.0, A. ABBYY 9.0 <http://finereader.abbyy.com/> [Access date]: 22/03/2015.
- [2] ABBYY. ABBYY Cloud OCR SDK [URL] :<http://ocrsdk.com/> [Access Date]: 25/06/2014.
- [3] AGGARWAL, J., AND RYOO, M. Human activity analysis. *ACM Computing Surveys* 43, 3 (Apr. 2011), 1–43.
- [4] AHONEN, T., HADID, A., AND PIETIKÄINEN, M. Face description with local binary patterns: application to face recognition. *IEEE transactions on pattern analysis and machine intelligence* 28, 12 (Dec. 2006), 2037–41.
- [5] ALLEN, J. F., AND FERGUSON, G. Actions and Events in Interval Temporal Logic. Tech. rep., Rochester, NY, USA, 1994.
- [6] AYACHE, S., QUÉNOT, G., AND GENSEL, J. Classifier Fusion for {SVM}-Based Multimedia Semantic Indexing. *Advances in Information Retrieval* 4425 (2007), 494–504.
- [7] BALLAN, L., BERTINI, M., BIMBO, A., SEIDENARI, L., AND SERRA, G. Event detection and recognition for semantic annotation of video. *Multimedia Tools and Applications* 51, 1 (Nov. 2010), 279–302.
- [8] BAY, H., ESS, A., TUYTELAARS, T., AND GOOL, L. V. Speeded-Up Robust Features (SURF).
- [9] BAY, H., TUYTELAARS, T., AND GOOL, L. V. SURF : Speeded Up Robust Features.
- [10] BELHUMEUR, P. N., CHEN, D., FEINER, S., JACOBS, D. W., KRESS, W. J., LING, H., LOPEZ, I., AND RAMAMOORTHY, R. Searching the World ’ s Herbaria : A System for Visual Identification of Plant Species. 116–129.
- [11] BOSWELL, D. Introduction to Support Vector Machines. 1–15.

- [12] BREIMAN, L. RANDOM FORESTS. 1–33.
- [13] CAO, Y., WANG, C., LI, Z., ZHANG, L., AND ZHANG, L. Spatial-bag-of-features. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), 3352–3359.
- [14] CHANG, S. F., SIKORA, T., AND PURI, A. Overview of the MPEG-7 standard. *IEEE Transactions on Circuits and Systems for Video Technology* 11, 6 (2001), 688–695.
- [15] CIRESAN, D. C., MEIER, U., GAMBARDELLA, L. M., AND SCHMIDHUBER, J. Deep Big Simple Neural Nets Excel on Handwritten Digit Recognition. 14.
- [16] COATES, A., CARPENTER, B., CASE, C., SATHEESH, S., SURESH, B., WANG, T., WU, D. J., AND NG, A. Y. Text Detection and Character Recognition in Scene Images with Unsupervised Feature Learning. *2011 International Conference on Document Analysis and Recognition* (Sept. 2011), 440–445.
- [17] COLUMBIA UNIVERSITY, UNIVERSITY OF MARYLAND, AND INSTITUTION, S. Leaf-snap [URL] : <https://itunes.apple.com/us/app/leafsnap/id430649829?mt=8> [Access Date]: 28/06/2014.
- [18] CSURKA, G., DANCE, C. R., FAN, L., WILLAMOWSKI, J., AND BRAY, C. Visual categorization with bags of keypoints. *Proceedings of the ECCV International Workshop on Statistical Learning in Computer Vision* (2004), 59–74.
- [19] DAILIANAS, A., ALLEN, R. B., NJ, M., AND ENGLAND, P. 1 INTRODUCTION 2 SEGMENTATION METHODS.
- [20] DALAL, N., AND TRIGGS, B. Histograms of Oriented Gradients for Human Detection. *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1* (2005), 886–893.
- [21] DE CAMPOS, T., BABU, B. R., AND VARMA, M. Character Recognition in Natural Images. *Proc Int Conf Computer Vision Theory and Applications* (2009).
- [22] EPSHTEIN, B., OFEK, E., AND WEXLER, Y. Detecting text in natural scenes with stroke width transform. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, d (June 2010), 2963–2970.
- [23] GEUSEBROEK, J. M., AND SMEULDERS, A. W. M. A six-stimulus theory for stochastic texture. *International Journal of Computer Vision* 62 (2005), 7–16.
- [24] GOMEZ, L., AND KARATZAS, D. Multi-script Text Extraction from Natural Scenes. *2013 12th International Conference on Document Analysis and Recognition* (Aug. 2013), 467–471.

- [25] GOOGLE. Google Docs OCR [URL] : https://developers.google.com/google-apps/documents-list/#uploading_documents_using_optical_character_recognition_ocr [Access Date]: 25/06/2014.
- [26] GREVET, C., CHOI, D., KUMAR, D., AND GILBERT, E. Overload is Overloaded : Email in the Age of Gmail. 793–802.
- [27] GURU, D. S., MANJUNATH, S., SHIVAKUMARA, P., AND TAN, C. L. An eigen value based approach for text detection in video. *Proceedings of the 8th IAPR International Workshop on Document Analysis Systems - DAS '10* (2010), 501–506.
- [28] HEINZ TSCHABITSCHER. How Many Email Users Are There? [URL] : http://email.about.com/od/emailtrivia/f/how_many_email.htm [Access Date]: 10/06/2014.
- [29] HENTSCHEL, C., BLÜMEL, I., AND SACK, H. Automatic Annotation of Scientific Video Material based on Visual Concept Detection. *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies - i-Know '13* (2013), 1–8.
- [30] HUANG, J., KUMAR, S. R., MITRA, M., ZHU, W. J., AND ZABIH, R. Spatial color indexing and applications. *International Journal of Computer Vision* 35, 3 (1999), 245–268.
- [31] HUANG, T. Linear spatial pyramid matching using sparse coding for image classification. *2009 IEEE Conference on Computer Vision and Pattern Recognition* (June 2009), 1794–1801.
- [32] IKEDA, H., MAEDA, M., KATO, N., AND KASHIMURA, H. Classification of human actions using face and hands detection. *Proceedings of the 12th annual ACM international conference on Multimedia - MULTIMEDIA '04* (2004), 484.
- [33] JAIN, A. K., DUIN, R. P. W., AND MAO, J. Statistical Pattern Recognition : A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 1 (2000), 4–38.
- [34] JAIN, A. K., AND FARROKHNI, F. Unsupervised texture segmentation using Gabor filters. *Systems, Man and Cybernetics, 1990. Conference Proceedings., IEEE International Conference on 000* (1990), 14–19.
- [35] JANSOHN, C., ULGES, A., AND BREUEL, T. M. Detecting pornographic video content by combining image features with motion information. *Proceedings of the seventeen ACM international conference on Multimedia - MM '09* (2009), 601.
- [36] JESUS, R., ABRANTES, A. J., AND CORREIA, N. Methods for automatic and assisted image annotation. *Multimedia Tools and Applications* 55, 1 (Aug. 2010), 7–26.

- [37] JONES, M. J. Robust Real-time Object Detection Paul Viola February 2001.
- [38] JORDAN, M., AND KLEINBERG, J. *Information Science and Statistics*, vol. 4. 2006.
- [39] KAEHLER, A., AND GARY, B. *Learning OpenCV*. O'Reilly Media, 2013.
- [40] KIM, J., GWON, R.-H., PARK, J.-T., KIM, H., AND KIM, Y.-S. A Semi-Automatic Video Annotation Tool to Generate Ground Truth for Intelligent Video Surveillance Systems. *Proceedings of International Conference on Advances in Mobile Computing & Multimedia - MoMM '13* (2013), 509–513.
- [41] KOLMOGOROV, V. Convergent tree-reweighted message passing for energy minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 10 (2006), 1568–1583.
- [42] KOTSIANTIS, S. B., ZAHARAKIS, I. D., AND PINTELAS, P. E. Machine learning: A review of classification and combining techniques. *Artificial Intelligence Review* 26 (2006), 159–190.
- [43] KUMAR, N., BELHUMEUR, P. N., BISWAS, A., JACOBS, D. W., KRESS, W. J., LOPEZ, I., AND SOARES, V. B. Leafsnap : A Computer Vision System for Automatic Plant Species Identification. 1–14.
- [44] LAAKSONEN, J., KOSKELA, M., AND OJA, E. PicSOM-self-organizing image retrieval with MPEG-7 content descriptors. *Neural Networks, IEEE ...* 13, 4 (2002), 841–853.
- [45] LAMEIRA, A. Real-Time Object Recognition using Mobile Devices. 687–690.
- [46] LE, Q. V. Building high-level features using large scale unsupervised learning. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (2013), 8595–8598.
- [47] LECUN, Y., BOSER, B., DENKER, J. S., HENDERSON, D., HOWARD, R. E., HUBBARD, W., AND JACKEL, L. D. Backpropagation Applied to Handwritten Zip Code Recognition, 1989.
- [48] LEE, J.-J., LEE, P.-H., LEE, S.-W., YUILLE, A., AND KOCH, C. AdaBoost for Text Detection in Natural Scene. *2011 International Conference on Document Analysis and Recognition* (Sept. 2011), 429–434.
- [49] LI, B., AND CHELLAPPA, R. Face verification through tracking facial features. *J. Opt. Soc. Am. A* 18, 12 (2001), 2969–2981.
- [50] LI, J., WU, W., WANG, T., AND ZHANG, Y. One step beyond histograms: Image representation using markov stationary features. *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2008).

- [51] LI, T., MEI, T., KWEON, I. S., AND HUA, X. S. Contextual bag-of-words for visual categorization. *IEEE Transactions on Circuits and Systems for Video Technology* 21, 4 (2011), 381–392.
- [52] LI, Z., FU, Y., HUANG, T., AND YAN, S. Real-time human action recognition by luminance field trajectory analysis. *Proceeding of the 16th ACM international conference on Multimedia - MM '08* (2008), 671.
- [53] LI HUIPING; DOERMANN, D.; KIA, O. Automatic text detection and tracking in digital video.
- [54] LIENHART, R., AND WERNICKE, A. Localizing and segmenting text in images and videos. *IEEE Transactions on Circuits and Systems for Video Technology* 12, 4 (Apr. 2002), 256–268.
- [55] LING, H., MEMBER, S., AND JACOBS, D. W. Shape Classification Using the Inner-Distance. 286–299.
- [56] LIU, C., WANG, C., AND DAI, R. Text detection in images based on unsupervised classification of edge-based features. *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)* (2005), 610–614 Vol. 2.
- [57] LOWE, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60, 2 (Nov. 2004), 91–110.
- [58] LUCAS, S. ICDAR 2005 text locating competition results. *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)* (2005), 80–84 Vol. 1.
- [59] LUCAS, S., PANARETOS, A., SOSA, L., TANG, A., WONG, S., AND YOUNG, R. ICDAR 2003 robust reading competitions. *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* 1 (2003), 682–687.
- [60] LUCAS, S. M., PANARETOS, A., SOSA, L., TANG, A., WONG, S., AND YOUNG, R. ICDAR 2003 robust reading competitions, 2003.
- [61] MANJUNATH, B., AND MA, W. Texture features for browsing and retrieval of image data. *\mbox{IEEE} Trans. on Pattern Analysis and Machine Intelligence* 8, 8 (1996), 837–842.
- [62] MAO, W., CHUNG, F.-L., LAM, K. K. M., AND SIU, W.-C. Hybrid Chinese/English Text Detection in Images and Video Frames. In *Proceedings of the 16th International Conference on Pattern Recognition (ICPR'02) Volume 3 - Volume 3* (Washington, DC, USA, 2002), ICPR '02, IEEE Computer Society, pp. 31015—.
- [63] MATAS, J., CHUM, O., URBAN, M., AND PAJDLA, T. Robust Wide Baseline Stereo from. In *British Machine Vision Conference* (2002), 384–393.

- [64] MATEUS, J., MALHEIRO, F., CAVACO, S., CORREIA, N., AND JESUS, R. Video annotation of TV content using audiovisual information. *2012 International Conference on Multimedia Computing and Systems* (May 2012), 113–118.
- [65] MATEUS, J. A. D. G. M. Extracção e Representação de Metadados num Contexto de Produção de Vídeo.
- [66] MISHRA, A. Top-down and Bottom-up Cues for Scene Text Recognition by Top-Down and Bottom-up Cues for Scene Text Recognition.
- [67] MISHRA, A., ALAHARI, K., AND JAWAHAR, C. Scene Text Recognition using Higher Order Language Priors. *Bmvc* (2012), 1–11.
- [68] NEUMANN, L., AND MATAS, J. A method for text localization and recognition in real-world images. 9–11.
- [69] NEUMANN, L., AND MATAS, J. Real-time scene text localization and recognition. ... *Vision and Pattern Recognition (CVPR)*, ... (2012).
- [70] NEUMANN, L., AND MATAS, J. Scene Text Localization and Recognition with Oriented Stroke Detection. *Iccv*, December (2013).
- [71] NOTE, A. The Basics of Colour Perception and Measurement. *Hunter Lab* (2012).
- [72] OCRWEBSERVICE. OCR Web Service [URL] : <http://www.ocrwebservice.com/Default.aspx> [Access Date]: 26/06/2014.
- [73] OMNIVORES, D. How Tablets , Smartphones and Connected Devices are Changing U . S . Digital Media Consumption Habits. 1–33.
- [74] PASS, G., ZABIH, R., AND MILLER, J. Comparing images using color coherence vectors. *Proceedings of the fourth ACM international conference on Multimedia (MULTIMEDIA '96)* (1996), 65–73.
- [75] PATIL, N. K., VASUDHA, S., AND BOREGOWDA, L. R. A Novel Method for Illumination Normalization for Performance Improvement of Face Recognition System. *2013 International Symposium on Electronic System Design* (Dec. 2013), 148–152.
- [76] PHAN, T. Q., SHIVAKUMARA, P., AND TAN, C. L. Detecting Text in the Real World. 765–768.
- [77] QI, G.-J., HUA, X.-S., RUI, Y., TANG, J., MEI, T., WANG, M., AND ZHANG, H.-J. Correlative multilabel video annotation with temporal kernels. *ACM Transactions on Multimedia Computing, Communications, and Applications* 5, 1 (Oct. 2008), 1–27.
- [78] RHODY, D. H. Finding the connected components in an image.

- [79] ROAD, Z. E. HORROR VIDEO SCENE RECOGNITION VIA MULTIPLE-INSTANCE LEARNING Jianchao Wang , Bing Li , Weiming Hu , Ou Wu. 1325–1328.
- [80] RYOO, M. S., AND AGGARWAL, J. K. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. *2009 IEEE 12th International Conference on Computer Vision, Iccv* (Sept. 2009), 1593–1600.
- [81] SAIDANE, Z., GARCIA, C., COURTEL, C., AND CESSON, S. Automatic Scene Text Recognition using a Convolutional Neural Network. 100–106.
- [82] SANDE, K. E. A. V. D., GEVERS, T., AND SNOEK, C. G. M. Evaluation of Color Descriptors for Object and Scene.pdf.
- [83] SCHULDT, C., LAPTEV, I., AND CAPUTO, B. Recognizing Human Actions : A Local SVM Approach. 3–7.
- [84] SHAHAB, A., SHAFAIT, F., AND DENGEL, A. ICDAR 2011 robust reading competition challenge 2: Reading text in scene images. *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR* (2011), 1491–1496.
- [85] SHAHAB, A., SHAFAIT, F., AND DENGEL, A. ICDAR 2011 Robust Reading Competition Challenge 2: Reading Text in Scene Images. *2011 International Conference on Document Analysis and Recognition* (Sept. 2011), 1491–1496.
- [86] SHARMA, N., PAL, U., AND BLUMENSTEIN, M. Recent advances in video based document processing: A review. *Proceedings - 10th IAPR International Workshop on Document Analysis Systems, DAS 2012* (2012), 63–68.
- [87] SHI, C., WANG, C., XIAO, B., ZHANG, Y., AND GAO, S. Scene text detection using graph model built upon maximally stable extremal regions. *Pattern Recognition Letters* 34, 2 (2013), 107–116.
- [88] SHI, C., WANG, C., XIAO, B., ZHANG, Y., GAO, S., AND ZHANG, Z. Scene text recognition using part-based tree-structured character detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2013), 2961–2968.
- [89] SHIVAKUMARA, P., DUTTA, A., TAN, C. L., AND PAL, U. Multi-oriented scene text detection in video based on wavelet and angle projection boundary growing. *Multimedia Tools and Applications* 72 (2014), 515–539.
- [90] SHIVAKUMARA, P., PHAN, T. Q., AND TAN, C. L. A Robust Wavelet Transform Based Technique for Video Text Detection. *2009 10th International Conference on Document Analysis and Recognition* 1 (2009), 1285–1289.

- [91] SMEULDERS, A. W. M., WORRING, M., SANTINI, S., GUPTA, A., AND JAIN, R. Content-based image retrieval at the end of the early years. *{IEEE} Transactions in Pattern Analysis and Machine Intelligence* 22, 12 (2000), 1349–1380.
- [92] SNOEK, C. G., AND WORRING, M. Concept-Based Video Retrieval. 215–322.
- [93] STATISTICS, I. U. INTERNET USAGE STATISTICS 2014 Q4.
- [94] STOLCKE, A. Srilm — an Extensible Language Modeling Toolkit. *System 2*, Denver, Colorado (2002), 901–904.
- [95] TIMM, F., AND BARTH, E. Accurate eye centre localisation by means of gradients. *International Conference on Computer Theory and Applications ({VISAPP})* (2011), 125–130.
- [96] TOPKARA, M., PAN, S., LAI, J., DIRIK, A. E., WOOD, S., AND BOSTON, J. “ You ’ ve G ot V ideo ”: Increasing Clickthrough W hen Sharing Enterprise Video with Email. 565–568.
- [97] TUYTELAARS, T., AND MIKOLAJCZYK, K. Local Invariant Feature Detectors: A Survey. 177–280.
- [98] VAN GEMERT, J. C., GEUSEBROEK, J. M., VEENMAN, C. J., SNOEK, C. G. M., AND SMEULDERS, A. W. M. Robust scene categorization by learning image statistics in context. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2006* (2006).
- [99] VEEDMEE. What’s Veedmee? [URL] : <http://veedmee.com/about/> [Access Date]: 10/06/2014.
- [100] VELTKAMP, R., AND HAGEDOORN, M. State of the Art in Shape Matching. *Principles of visual information ...* (2001).
- [101] VIOLA, P., AND JONES, M. Rapid object detection using a boosted cascade of simple features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001 1* (2001).
- [102] WANG, J., YANG, J., YU, K., LV, F., HUANG, T., AND GONG, Y. Locality-constrained Linear Coding for image classification. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (June 2010), 3360–3367.
- [103] WANG, K., BABENKO, B., AND BELONGIE, S. End-to-end scene text recognition. *Proceedings of the IEEE International Conference on Computer Vision*, 4 (2011), 1457–1464.
- [104] WANG, K., AND BELONGIE, S. Word spotting in the wild. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6311 LNCS, PART 1 (2010), 591–604.

- [105] WANG, L., FAN, W., HE, Y., AND SUN, J. Fast and Accurate Text Detection in Natural Scene Images with User-intention. 2920–2925.
- [106] WANG, M., HUA, X. S., HONG, R., TANG, J., QI, G. J., AND SONG, Y. Unified video annotation via multigraph learning. *IEEE Transactions on Circuits and Systems for Video Technology* 19, 5 (2009), 733–746.
- [107] WARFIELD, N. I. W., K, S., WEISENFELD, N. I., AND WARFIELD, S. K. Kernel Codebooks for Scene Categorization. *NeuroImage* 6893 (2011), 1–8.
- [108] WECHSLER, H., KAKKAD, V., HUANG, J., GUTTA, S., AND CHEN, V. Automatic Video-based Person Authentication Using the RBF Network. In *Proceedings of the First International Conference on Audio- and Video-Based Biometric Person Authentication* (London, UK, UK, 1997), AVBPA '97, Springer-Verlag, pp. 85–92.
- [109] WONG, E. K., AND CHEN, M. A new robust algorithm for video text extraction. *Pattern Recognition* 36, 6 (June 2003), 1397–1406.
- [110] XU, C., AND PRINCE, J. L. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing* 7, 3 (1998), 359–369.
- [111] YAMATO, J. ; NTT HUMAN INTERFACE LABS., YOKOSUKA, JAPAN ; JUN OHYA ; ISHII, K. Recognizing human action in time-sequential images using hidden Markov model. *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR '92., 1992 IEEE Computer Society Conference on*, 379 – 385.
- [112] YE, Q., AND DOERMANN, D. Scene text detection via integrated discrimination of component appearance and consensus. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8357 LNCS (2014), 47–59.
- [113] YE, Q., HUANG, Q., GAO, W., AND ZHAO, D. Fast and robust text detection in images and video frames. *Image and Vision Computing* 23, 6 (June 2005), 565–576.
- [114] YIN, X., HUANG, K., AND HAO, H.-W. Robust text detection in natural scene images. 1–10.
- [115] YOKOBAYASHI, M., AND WAKAHARA, T. Binarization and Recognition of Degraded Characters Using a Maximum Separability Axis in Color Space and GAT Correlation. *18th International Conference on Pattern Recognition (ICPR'06)* (2006), 885–888.
- [116] YU, X., WANG, J., KAYS, R., JANSEN, P. A., WANG, T., AND HUANG, T. Automated identification of animal species in camera trap images. *EURASIP Journal on Image and Video Processing* 2013, 1 (2013), 52.

- [117] YUAN, J., WANG, H., XIAO, L., ZHENG, W., LI, J., LIN, F., AND ZHANG, B. A formal study of shot boundary detection. *IEEE Transactions on Circuits and Systems for Video Technology* 17, 2 (2007), 168–186.
- [118] ZHANG, C., AND ZHANG, Z. A Survey of Recent Advances in Face Detection.
- [119] ZHANG, J., AND KASTURI, R. Extraction of Text Objects in Video Documents: Recent Progress. *2008 The Eighth IAPR International Workshop on Document Analysis Systems* (Sept. 2008), 5–17.
- [120] ZHAO, W., AND ROSENFELD, A. Face Recognition : A Literature Survey AND. 399–458.
- [121] ZHU, X., AND RAMANAN, D. Face detection, pose estimation, and landmark localization in the wild. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2012), 2879–2886.